Black-Box Constructions of Protocols for Secure Computation*

Iftach Haitner† Yuval Ishai‡ Eyal Kushilevitz† Yehuda Lindell§
Erez Petrank†

May 17, 2010

Abstract

It is well known that secure computation without an honest majority requires computational assumptions. An interesting question that therefore arises relates to the way such computational assumptions are used. Specifically, can the secure protocol use the underlying primitive (e.g., a one-way trapdoor permutation) in a *black-box* way, treating it as an oracle, or must it be *nonblack-box* (by referring to the code that computes the primitive)? Despite the fact that many general constructions of cryptographic schemes refer to the underlying primitive in a black-box wayonly, there are some constructions that are inherently nonblack-box. Indeed, all known constructions of protocols for general secure computation that are secure in the presence of a malicious adversary and without an honest majority use the underlying primitive in a nonblack-box way (requiring to prove in zero-knowledge statements that relate to the primitive).

In this paper, we study whether such nonblack-box use is essential. We answer this question in the negative. Concretely, we present a *fully black-box reduction* from oblivious transfer with security against malicious parties to oblivious transfer with security against semi-honest parties. As a corollary, we get the first constructions of general multiparty protocols (with security against malicious adversaries and without an honest majority) which only make a *black-box* use of semi-honest oblivious transfer, or alternatively a black-box use of lower-level primitives such as enhanced trapdoor permutations or homomorphic encryption.

Keywords: Theory of cryptography, secure computation, black-box reductions, oblivious transfer

*This paper combines the results appearing in [17] and [22]. The last four authors were supported by grant 36/03 from the Israel Science Foundation.

†Microsoft Research, New England Campus. email: iftach@microsoft.com. Most of this work was performed while at the Weizmann Institute of Science.

‡Department of Computer Science, Technion, Israel. email: {yuvali,eyalk,erez}@cs.technion.ac.il

§Department of Computer Science, Bar-Ilan University, Israel. email: lindell@cs.biu.ac.il. Much of this work was carried out while the author was visiting the Technion.

1 Introduction

It is a known fact that most cryptographic tasks require the use of computational hardness assumptions. These assumptions typically come in two types: *specific assumptions* like the hardness of factoring, RSA, discrete log and others, and *general assumptions* like the existence of one-way functions, trapdoor permutations and others. In this paper, we refer to general assumptions and how they are used. Specifically, we consider an intriguing question regarding how secure protocols utilize a primitive that is assumed to carry some hardness property. Here again, there is a clear distinction between two types of uses:

1. **Black-box usage:** a protocol (or construction) uses a primitive in a black-box way if it refers only to the input/output behavior of the primitive.[1] For example, if the primitive is a trapdoor permutation, then the protocol may sample a permutation and its domain, and may compute the permutation and its inverse (if the trapdoor is given). Beyond this, no reference is made to the primitive. In particular, the *code* used to compute the permutation (or carry out any other task) is not referred to by the protocol. The vast majority of constructions in cryptography are of this black-box type.

2. **Nonblack-box usage:** a protocol (or construction) uses a primitive in a nonblack-box way if it refers to the code for computing its functionality. A typical example of a nonblack-box construction is where a *Karp reduction* is applied to the circuit computing the function, say, in order to prove an $\mathcal{NP}$ zero-knowledge proof, as in [15].

A rich and fruitful body of work, initiated in [21], attempts to draw the borders between possibility and impossibility for black-box constructions in cryptography. While many of the relations between primitives are well understood, there are still some important tasks for which the *only* constructions that we have rely on nonblack-box access to the assumed primitive, yet the existence of a black-box construction is not ruled out. In particular, this is the case for all constructions of public-key encryption schemes that are secure against chosen-ciphertext attacks [8, 37, 30]. More relevant to this work is the fact that *all* known general constructions of multiparty protocols that are secure without an honest majority and in the presence of malicious adversaries, originating from [16], use nonblack-box access to the assumed primitive.[2] (We note that by "general constructions", we mean construction that can be used to securely compute any functionality.)

The above phenomenon begs the following informally-stated question:

> *Is it possible to construct general protocols for secure computation without an honest majority and with malicious adversaries, given only* black-box access *to a "low-level" primitive?*

[1] It is typically also required that the *security proof* of the construction be black-box in the sense that an adversary breaking the protocol can be used as an oracle in order to break the underlying primitive. In such a case the construction is referred to as a *fully black-box reduction*. All of the constructions in this paper are in fact fully black-box reductions. See Section 2.3 and [11, 12, 36] for further discussion of reductions in cryptography.

[2] We stress that the above discussion is only true when considering general assumptions. Furthermore, it is only true when considering "low-level primitives" like trapdoor permutations. Specifically, there *do* exist constructions of secure multiparty protocols that use only black-box access to an oblivious transfer protocol with security against malicious parties [25]. However, since it is not known how to construct the latter oblivious transfer protocols using only black-box access to, say trapdoor permutations, the overall construction obtained does not use its "low-level" primitive in a black-box way.

Answering the above question is of interest for the following reasons. First, it is of theoretical interest to understand whether or not nonblack-box access to a primitive is necessary for these tasks. An answer to this question would enhance our understanding of how hardness assumptions can (or must) be used. Second, as we have mentioned, the nonblack-box use of the underlying primitive is typically utilized in order to apply a Karp reduction for the purpose of using a (general) zero-knowledge proof. Such reductions are highly inefficient and are unlikely to be very useful in practice. Furthermore, in these protocols the *communication complexity* depends on the complexity of computing the primitive and the *computational complexity* grows more than linearly with that of the primitive. (An exception to this rule is the communication-efficient compiler presented in [33], which relies on the communication-efficient arguments of [27, 31]. However, the *computational* complexity of the protocol of [33] is even worse than the GMW protocol [16].)

To illustrate the type of inefficiency resulting from current nonblack-box constructions, consider the following hypothetical scenario. Suppose that, due to major advances in cryptanalytic techniques, the security parameter must be large enough so that *all* basic cryptographic primitives require a full second of computation on a fast CPU. In such a case, would it still be possible to carry out a distributed task like oblivious transfer? Current nonblack-box techniques (e.g., the GMW protocol [16]) require parties to prove in zero-knowledge statements that involve the computation of the underlying primitive, say a trapdoor permutation. These zero-knowledge protocols, in turn, invoke cryptographic primitives for any gate of a circuit computing a trapdoor permutation. Since (by our assumption) a trapdoor permutation takes one second to compute, its circuit implementation contains trillions of gates, thereby requiring the protocol trillions of second to run. In contrast, a black-box construction of oblivious transfer from the trapdoor permutation primitive would make the number of invocations of the primitive independent of the complexity of implementing the primitive, thus making oblivious transfer feasible even in the hypothetical scenario described above.

We conclude that the current nonblack-box use of the underlying primitives constitutes an obstacle to efficiency. It is therefore of great interest to know whether or not it is possible to obtain solutions to these tasks that do not suffer from this obstacle. (We note that the inefficiency of nonblack-box constructions here is quite ironic because in many areas of cryptography, black-box constructions have been shown to have inherent computational limitations [28, 10].) Despite the above, we stress that the focus of this paper is not on efficiency, but rather on the *theoretical question* of whether or not it is possible to obtain the aforementioned black-box constructions. We believe this question to be interesting in its own right.

Our results. We show how to construct protocols for general secure multiparty computation (for the case of no honest majority and malicious adversaries), given black-box access to any oblivious transfer protocol with security against *semi-honest* adversaries. Using previous constructions, the latter oblivious transfer protocol can be based (in a black-box way) on either homomorphic encryption schemes (cf. [1]) or enhanced trapdoor permutations (cf. [13]). We note that the standard general constructions of secure computation protocols from "low-level" primitives rely on either enhanced trapdoor permutations or homomorphic encryption schemes. However, all previous protocols used these primitives in an inherently nonblack-box way. This was the case even for protocols that implement very simple functionalities, such as oblivious transfer [35, 9]. More concretely, we prove the following:

Theorem 1.1 *There exist protocols for securely computing any multiparty functionality without*

an honest majority and in the presence of static malicious adversaries, that rely only on black-box access to a semi-honest*-secure oblivious transfer protocol, to a family of enhanced trapdoor permutations, or to a homomorphic encryption scheme.*

We remark that nonblack-box access is not typically used when considering semi-honest adversaries [39, 16]. Rather, the nonblack-box access is utilized in known protocols in order to have the parties prove (in zero-knowledge) that they are correctly following the protocol specification. This is necessary for preventing a malicious adversary from effectively deviating from the protocol instructions. We note also that in the case of an honest majority, it is possible to securely compute any functionality information-theoretically, and without any hardness assumption [2, 5]. Thus, no primitive at all is needed. For this reason, we focus on the case of no honest majority (including the important two-party case) and malicious adversaries.

Techniques. We prove Theorem 1.1 by showing that it is possible to construct an oblivious transfer protocol that is secure in the presence of malicious adversaries given only black-box access to an oblivious transfer protocol that is secure in the presence of semi-honest adversaries. Our construction is *fully black-box* meaning that the protocol that is secure in the presence of malicious adversaries uses only black-box access to the semi-honest protocol, and our proof of security shows the existence of an adversary that breaks the semi-honest protocol given black-box access to any adversary that breaks the malicious protocol. Formally, we prove the following:

Theorem 1.2 *The exists a fully black-box reduction from oblivious transfer that is secure in the presence of malicious adversaries to oblivious transfer that is secure in the presence of semi-honest adversaries.*

In order to prove Theorem 1.2, we begin by constructing oblivious transfer protocols that use only black-box access to semi-honest oblivious transfer, but provide rather weak security guarantees. We then "boost" the security of these protocols in order to obtain protocols that are secure in the presence of malicious adversaries. Previous constructions that have followed this paradigm apply general zero-knowledge proofs to have the parties prove that they are behaving according to the specification of the semi-honest protocol [16]. Importantly, these proofs employ a Karp reduction on the protocol specification, they are not black-box. Since we wish to make our construction black-box, we take a different route. Specifically, we begin by introducing the notion of a defensible adversary. In order to describe this notion, we describe what a defense is: a defense is an input and random-tape that is provided by the adversary *after* the protocol execution concludes. A defense is good if the honest party upon that input and random-tape would have sent the same messages as the adversary sent. Such a defense is a supposed "proof" of honest behavior. However, the adversary need not actually behave honestly and can construct its defense retroactively (after the execution concludes). A protocol is said to be private in the presence of defensible adversaries if privacy is preserved in the event that an adversary provides a good defense. However, in the case that the adversary doesn't provide a good defense, *nothing is guaranteed*, and the entire honest party's input may be learned. This notion is therefore rather weak. We note that known oblivious transfer protocol like that of [9] are not secure under this notion. The first step of our proof is therefore a proof that any semi-honest oblivious transfer protocol can be efficiently modified – in a black-box way – into one that is secure under this notion. Next, we show that it is possible to construct oblivious transfer that is secure in the presence of malicious adversaries from oblivious

transfer that is private in the presence of defensible adversaries. Once again, this construction is fully black-box.

We now give a more detailed overview of the construction. Recall that the first step is constructing oblivious transfer protocols that are private in the presence of *defensible adversaries* from any oblivious transfer that is secure in the presence of semi-honest adversaries. This construction is essentially a very degenerate version of the GMW compiler [16]. Specifically, it works by having the parties run a coin-tossing and input commitment protocol. The parties are then supposed to run the protocol using the committed inputs and coins. The defense is then just a decommitment; observe that given such a decommitment it is possible to check that the parties behaved in a semi-honest manner, as required. We remark that our transformation from semi-honest security to defensible security is *generic* and works for *all* protocols (and not just for oblivious transfer).

Having obtained oblivious transfer that is secure for defensible adversaries, we use this to construct a new oblivious transfer protocol that is still private in the presence of defensible senders, but is *secure* in the presence of malicious receivers (where security is "full security" according to the ideal/real simulation paradigm). This is achieved using the so-called *cut-and-choose* technique. That is, many oblivious transfer executions (using random inputs) are run, and the receiver is asked to present a defense for its behavior in half of them. If it indeed presents good defenses, then we are guaranteed that it behaved somewhat honestly in most of the executions. Finally, the oblivious transfer executions are *combined* into a single instance of oblivious transfer which has the stronger security guarantee.

We stress that much of the difficulty in implementing and analyzing the above step comes from the requirement that the protocol be secure according to the ideal/real simulation paradigm, rather than just private. Indeed, some efficient protocols for oblivious transfer from the literature [34, 1, 24] are private for both (malicious) parties, but are not fully secure for either party. Nevertheless, we are able to boost both the type of adversary resisted by the protocol (from a defensible to a malicious adversary) and its security guarantee (from privacy to full simulation-based security).

Next, we "reverse" the oblivious transfer protocol (i.e., by switching the sender and receiver roles) in order to obtain a protocol with reversed security properties. Specifically, this next protocol is secure in the presence of malicious senders and private in the presence of defensible receivers. At this point, we reapply our security boosting technique in order to obtain a protocol that is "fully secure"; that is, a protocol that is secure in the presence of malicious senders and receivers. See Figure 1 for the series of oblivious transfer protocols that we construct. Needless to say, each protocol uses its subprotocol in a black-box way.

Protocol number	**Security for corrupted sender**	**Security for corrupted receiver**
3.3	Private for defensible sender	Private for defensible receiver
4.2	Private for defensible sender	Secure for malicious receiver
5.1	Secure for malicious sender	Private for defensible receiver
In Theorem 6.1	Secure for malicious sender	Secure for malicious receiver

Figure 1: The progression of our constructions: each protocol uses the previous one as a subprotocol.

We note that Protocol 3.3 uses any semi-honest oblivious transfer as a starting point and works for any protocol problem and not just oblivious transfer. This is in contrast to our other constructions that are all specifically tailored to oblivious transfer.

This completes our high-level description of how we prove Theorem 1.2. Once we have constructed secure oblivious transfer protocols that use only black-box access to primitives, we apply the well-known result of Kilian [25, 26] (alternatively, the recent protocol from [23]) which shows that any two-party functionality can be securely computed in the presence of malicious adversaries using black-box access to an oblivious transfer protocol that is secure in the presence of malicious adversaries. This therefore yields Theorem 1.1, as desired.

Related and subsequent work. In [7] it was shown that there are *constant-round* protocols for the setting of an *honest majority*, that use only black-box access to the assumed primitive (a pseudorandom generator). As we have mentioned, in the setting of an honest majority, it is possible to construct information-theoretically secure protocols (which are, by triviality, black-box). Nevertheless, there are no known (general) constant-round protocols for the information-theoretic setting, and so [7] relates to this issue. The techniques used in [7] and here are vastly different, due to the inherent differences between the setting of an honest majority and that of no honest majority.

In a subsequent work [6], it is shown that if an *ideal* commitment is available, then the results of the present work can be extended to provide universal composability and adaptive security (the latter assumes that the semi-honest oblivious transfer is adaptively secure).

In another subsequent work [23] it is shown that, building on our black-box reduction from malicious oblivious transfer to semi-honest oblivious transfer, it is possible to obtain a *constant rate* reduction of the same type. More concretely, n instances of oblivious transfer with security against malicious parties can be implemented by making a black-box use of only $O(n)$ instances of oblivious transfer with security against semi-honest parties (plus an *additive* term that depends polynomially on a statistical security parameter but does not depend on n).

2 Definitions

2.1 Preliminaries

A function $\epsilon(\cdot)$ is said to be *negligible* if $\epsilon(n) < n^{-c}$ for any constant $c > 0$ and sufficiently large n. A *distribution ensemble* $\{X_n\}_{n\in\mathsf{N}}$ is an infinite sequence of distributions, where the support of each distribution X_n consists of binary strings of some fixed length $m(n)$. We say that two distribution ensembles $\{X_n\}_{n\in\mathsf{N}}$ and $\{Y_n\}_{n\in\mathsf{N}}$, are *computationally indistinguishable*, and write $X_n \stackrel{\mathrm{c}}{\equiv} Y_n$, if for every (non-uniform) polynomial-size circuit family $\{A_n\}$, the distinguishing advantage $|\Pr[A_n(X_n) = 1] - \Pr[A_n(Y_n) = 1]|$ is negligible. Similarly, we say that the two distributions are *statistically indistinguishable* if the above holds for arbitrary (computationally unbounded) distinguishers A_n.

We will refer to standard cryptographic primitives such as one-way functions and commitment schemes. For self-containment, we include formal definitions of these primitives in Appendix A.

We will consider two-party protocols defined by a pair of probabilistic polynomial time algorithms P_1 and P_2. We denote by $\langle P_1(1^n, x_1, \rho_1), P_2(1^n, x_2, \rho_2)\rangle$ the transcript of an execution between parties P_1 and P_2 with a security parameter n, where P_i has input x_i and random-tape ρ_i. (Such a transcript consists of the messages exchanged between the two parties.) For brevity, we will sometimes omit the security parameter 1^n. The message sent by party P_i (on the above inputs) after having received the sequence of incoming messages α is denoted by $P_i(x_i, \rho_i; \alpha)$.

Stated otherwise, $P_i(x_i, \rho_i; \cdot)$ denotes the next message function of P_i. We denote the output of P_i in an execution by $\mathsf{output}_{P_i}\langle P_1(x_1, \rho_1), P_2(x_2, \rho_2)\rangle$ and its view (including its input, random-tape and received messages) by $\mathsf{view}_{P_i}\langle P_1(x_1, \rho_1), P_2(x_2, \rho_2)\rangle$. We denote by $\mathsf{output}_{P_i}\langle P_1(x_1), P_2(x_2)\rangle$ and $\mathsf{view}_{P_i}\langle P_1(x_1), P_2(x_2)\rangle$ the corresponding random variables when ρ_1 and ρ_2 are uniformly distributed. Finally, we denote by U_k the random variable that receives a uniformly distributed string in $\{0,1\}^k$.

2.2 Secure Computation

We assume readers to have basic familiarity with standard simulation-based definitions of secure two-party computation in a standalone setting. We provide a self-contained definition for completeness, and refer to [13, Chapter 7] for a fuller treatment.

Overview. In this work, we consider both *malicious adversaries,* who may arbitrarily deviate from the protocol specification, and *semi-honest adversaries,* who follow the protocol specification but may try to infer additional information from the messages they receive. We restrict the attention to *static corruptions* (meaning that the set of corrupted parties is fixed throughout the protocol's execution). We restrict our protocols to have *black-box simulators*, which treat the adversary as an oracle, and allow the simulators to run in *expected polynomial time* (counting each oracle call to the adversary as a single step). Finally, we assume adversaries to be *non-uniform* polynomial-time algorithms (or circuit families) and therefore, without loss of generality, assume that they are deterministic. However, we note that this is not essential and all of our proofs hold for the uniform model of computation.

Definitions. We let n denote a security parameter which is given as input to both parties. For simplicity we assume here that the private input of each party has a *fixed* length k, which does not grow with n. The security of a protocol is defined with respect to a *functionality* f. A two-party functionality is a (possibly randomized) mapping of a pair of inputs (x_1, x_2) to a pair of outputs (y_1, y_2). A two-party protocol for computing f is a protocol running in polynomial time and satisfying the following correctness requirement: For any pair of inputs (x_1, x_2), the pair of outputs (y_1, y_2) obtained in the end of an honest execution of the protocol is statistically indistinguishable from $f(x_1, x_2)$.

The security of a protocol (with respect to a functionality f) is defined by comparing the *real-world* execution of the protocol with an *ideal-world* evaluation of f by a trusted party. More concretely, it is required that for every adversary $\mathcal{A}$, which attacks the real execution of the protocol by corrupting one of the two parties, there exists an adversary $\mathcal{A}'$, also referred to as a *simulator*, which can "achieve the same effect" by corrupting the same party in an ideal evaluation of f. As noted above, we make the stronger requirement that there exist a single universal simulator Sim which has black-box access to $\mathcal{A}$. This is made more precise in what follows.

THE REAL EXECUTION. Let π be a protocol computing a two-party functionality f. Let $\mathcal{A} = \{\mathcal{A}_n\}$ be an efficient nonuniform adversary which plays the role of either P_1 or P_2. For a security parameter n, inputs $x_1, x_2 \in \{0,1\}^k$ and party index $i \in \{1,2\}$, we let $\text{REAL}_{\pi,\mathcal{A},i}(n, x_1, x_2)$ be a random variable containing the identity i of the corrupted party, the view of the adversary when playing the role of P_i (and interacting with an uncorrupted P_{2-i} on security parameter n and inputs x_1, x_2), and the

output of the uncorrupted party P_{2-i}. That is,

$$\text{REAL}_{\pi,\mathcal{A},1}(n,x_1,x_2) = (1, \mathsf{view}_{\mathcal{A}_n}\langle \mathcal{A}_n(1^n,x_1), P_2(1^n,x_2,\rho_2)\rangle, \mathsf{output}_{P_2}\langle \mathcal{A}_n(1^n,x_1), P_2(1^n,x_2,\rho_2)\rangle),$$

where ρ_2 is uniformly distributed, and similarly

$$\text{REAL}_{\pi,\mathcal{A},2}(n,x_1,x_2) = (2, \mathsf{view}_{\mathcal{A}_n}\langle P_1(1^n,x_1,\rho_1), \mathcal{A}_n(1^n,x_2)\rangle, \mathsf{output}_{P_1}\langle P_1(1^n,x_1,\rho_1), \mathcal{A}_n(1^n,x_2)\rangle).$$

THE IDEAL EXECUTION. In the ideal execution, an adversary $\mathcal{A}'$ interacts with a trusted party who computes f on inputs provided by the two parties. More concretely, the ideal execution proceeds as follows:

- The uncorrupted party sends its real input to the trusted party. The adversary $\mathcal{A}'$ may send an arbitrary input on behalf of the corrupted party, depending on the real input and the security parameter. (Any missing or "invalid" value is substituted by a default value.) Denote by x_i' the value sent by party P_i.
- The trusted party computes $f(x_1', x_2') = (y_1, y_2)$. (If f is randomized, this computation involves random coins that are generated by the trusted party.) It then sends to the adversary its own output.
- The adversary chooses whether to continue or abort; this can be formalized by having the adversary send either a $\mathsf{continue}$ or abort message to the trusted party. In the former case, the trusted party sends to the uncorrupted party P_i its output value y_i. In the latter case, the trusted party sends the special symbol $\perp$ to the uncorrupted party.

After the above, $\mathcal{A}'$ outputs some (randomized) function of its view, and the uncorrupted party outputs the value received from the trusted party. We let $\text{IDEAL}_{f_\perp,\mathcal{A}',i}(n,x_1,x_2)$ be the random variable containing the identity i of the corrupted party, the output of the adversary in the end of the above process when playing the role of P_i (and interacting with an uncorrupted P_{2-i} on security parameter n and inputs x_1, x_2), and the output of the uncorrupted party P_{2-i}. The subscript "$\perp$" indicates that the adversary has the ability to abort the ideal execution.

DEFINING SECURITY. With the above in place, we can now define our strongest notion of security.

Definition 2.1 (Security against malicious parties) *Let π be a two-party protocol for computing a functionality f. We say that π $\mathsf{securely\ computes}$ f $\mathsf{in\ the\ presence\ of\ malicious\ adversaries}$ if there exists a probabilistic, expected polynomial-time (black-box) simulator* Sim *such that for every polynomial-size circuit family $\mathcal{A} = \{\mathcal{A}_n\}$, every choice $i \in \{1,2\}$ of corrupted party, and every choice of inputs $x_1, x_2 \in \{0,1\}^k$, we have*

$$\text{REAL}_{\pi,\mathcal{A},i}(n,x_1,x_2) \stackrel{\text{c}}{\equiv} \text{IDEAL}_{f_\perp,\mathcal{A}',i}(n,x_1,x_2),$$

where $\mathcal{A}' = \{\mathcal{A}_n'\}$ is the ideal execution adversary defined by $\mathcal{A}_n' = \mathsf{Sim}^{\mathcal{A}_n}(1^n)$.

A definition of security against *semi-honest* parties can be obtained by restricting the adversaries $\mathcal{A}$ and $\mathcal{A}'$ to be semi-honest. In the case of $\mathcal{A}$, this means that all messages sent by the adversary are computed according to π. (For implementing the honest strategy, $\mathcal{A}$ should receive an additional

random input ρ_i.) In the case of $\mathcal{A}'$ this means that $\mathcal{A}'$ must send the true input to the trusted party and never abort.

We note that the above definition does not allow the choice of inputs (x_1, x_2) to depend on the security parameter. Since we assume the inputs to come from a fixed domain, this is equivalent to the standard (and generally more stringent) definition which requires indistinguishability even when the choice of inputs (x_1, x_2) may depend on n.

We will mostly be interested in functionalities which deliver an output to only one of the two parties. In such a case, any protocol satisfying the above notion of "security with abort" can be easily turned into a similar protocol with full security (without abort). This is done by having the uncorrupted party handle an event where the protocol aborts by computing its output on its own (using a default value for the input of the uncorrupted party). Thus, for such functionalities we will use a simpler form of the above definition which does not allow the adversary to abort in the ideal execution.

2.3 Black-Box Reductions

A *reduction* from a primitive P to a primitive Q (sometimes referred to as a *construction* of primitive P using a primitive Q) consists of showing that if there exists a secure implementation C of Q, then there exists a secure implementation M_C of P. This is equivalent to showing that for every adversary that breaks M_C, there exists an adversary that breaks C. Such a reduction is *semi black-box* if the implementation M_C of P and the adversary breaking Q ignore the internal structure of Q's implementation and only use it as an oracle. The reduction is *fully black-box* if the proof of security is black box as well. That is, the adversary breaking Q ignores the internal structure of both Q's implementation and the (alleged) adversary breaking[3] P. One can similarly define a fully black-box reduction from P to a pair of primitives Q_1, Q_2; here it should be possible to use an adversary breaking P as an oracle for breaking *either* Q_1 or Q_2. A taxonomy of black-box reductions was provided by [36], and the reader is referred to this paper for a more complete and formal overview of these notions. All the reductions considered in this paper are fully black-box, though the black-box construction of an adversary breaking Q from an adversary breaking P will only be implicit in the security analysis.

2.4 Defensible Adversarial Behavior

We introduce the notion of defensible adversarial behavior. Loosely speaking, an adversary that exhibits defensible behavior may arbitrarily deviate from the protocol specification. However, at the *conclusion* of the protocol execution, the adversary must be able to justify or *defend* its behavior by presenting an input and a random-tape such that the honest party (with this input and random-tape) would behave in the same way as the adversary did. A protocol is "private" under defensible adversarial behavior if it is "private" whenever the adversary can successfully prevent such a good defense. We stress that if an adversary behaves maliciously and cannot provide a good defense, then no security guarantees are given.

[3]The meaning of "breaking" a primitive is usually clear from the security definition of the primitive. When P is a secure computation protocol with a black-box simulator Sim, as in Definition 2.1, an adversary breaking P consists of a real-execution adversary $\mathcal{A}$ attacking the protocol along with a distinguisher violating the indistinguishability condition between REAL and IDEAL from Definition 2.1.

The above notion of privacy against defensible adversaries will be made more formal later. For now, we formally define the notion of a *good defense*. Intuitively, a defense is an "explanation" of an adversary's behavior during the protocol execution. Such an explanation consists of an input and random-tape, and the defense is "good" if an honest party, given that input and random-tape, would have sent the same messages as the adversary did during the protocol execution. The formal definition follows.

Definition 2.2 (good defense for t): *Let t be the transcript of an execution of a protocol $\pi = (P_1, P_2)$ between an adversary $\mathcal{A}$ (say, controlling P_1) and the honest party (say P_2). Then, we say that the pair (x_1, ρ_1) constitutes a* good defense by $\mathcal{A}$ for t in π, *denoted $(x_1, \rho_1) = \mathsf{defense}^{\pi}_{\mathcal{A}}(t)$, if for every $\ell \in \mathsf{N}$ it holds that $\mathsf{sent}^{\mathcal{A}}_{\ell}(t) = P_1(x_1, \rho_1; \mathsf{received}^{\mathcal{A}}_{1,\ldots,\ell-1}(t))$, where $\mathsf{sent}^{P_i}_{\ell}(t)$ denotes the ℓ^{th} message sent by P_i in t and $\mathsf{received}^{P_i}_{1,\ldots,\ell}(t)$ denote the first ℓ messages received by P_i in t.*

In other words, every message sent by $\mathcal{A}$ in the execution is such that the honest party P_1 with input (x_1, ρ_1) would have sent the same message.

2.5 Security of Oblivious Transfer Protocols

An oblivious transfer protocol [35, 9] is a secure protocol for the following oblivious transfer functionality. Instead of P_1 and P_2 we refer to the two parties as a *sender* S and a *receiver* R. The sender's input consists of a pair of strings (s_0, s_1) and the receiver's input is a single selection bit $r \in \{0, 1\}$. The receiver's output is the selected string s_r, and the sender has no output. Thus, an oblivious transfer protocol has the property that the sender learns nothing about the receiver's selection bit r and the receiver obtains s_r, but learns nothing about s_{1-r}. (The variant of oblivious transfer that we use here is usually referred to as "1-out-of-2 OT" [9].) We will typically restrict the attention to the case of *bit oblivious transfer*, in which each of s_0 and s_1 is a single bit. Thus, by default, oblivious transfer should be interpreted as bit oblivious transfer.

As outlined in the introduction, we apply a sequence of black-box constructions to obtain an oblivious transfer protocol with security against malicious parties from any oblivious transfer protocol with security against semi-honest parties. The first step is a black-box construction of an oblivious transfer protocol that is private in the presence of a *defensible* adversary from an oblivious transfer protocol that is secure against semi-honest parties.

In the following we define both of the above notions of oblivious transfer. While a definition of semi-honest oblivious transfer can be derived as a special case of the general definition from Section 2.2, it will be convenient to use the more explicit (but equivalent) formulation give below.

Non-trivial oblivious transfer protocols. One technicality that must be dealt with is that a protocol that does nothing is trivially "private" in that it does not reveal anything about the parties' inputs. Of course, such a protocol is also useless. In order to make sure that the oblivious transfer protocols that we construct are "useful", we define the notion of a non-trivial oblivious transfer protocol. Such a protocol has the property that if both the sender and receiver are uncorrupted, then the receiver will receive its output as designated by the oblivious transfer functionality except, perhaps, with negligible probability in n. From here on, whenever we refer to an oblivious transfer protocol we implicitly make the above non-triviality requirement.

Privacy in the presence of semi-honest adversaries. In an oblivious transfer protocol, the sender is not supposed to learn anything about the receiver's input, and the receiver is supposed to learn only one of the sender's inputs and nothing about the other. The standard definition of *security* in the presence of semi-honest adversaries requires the existence of a simulator that receives the corrupted party's input and output, and generates a view that is indistinguishable from the view of the corrupted party (adversary) in a real protocol execution; see Section 2.2. Here we define a seemingly weaker notion that refers to "privacy with respect to a function g", and requires that a semi-honest adversary cannot distinguish between $g(x, y)$ and $g(x, y')$ where x is the adversary's input, y is a uniformly distributed input of the honest party used in the actual execution, and y' is a uniform input distributed independently of y. Of course, it is only possible to achieve privacy with respect to a function g if the designated output of the adversary does not enable it to distinguish between $g(x, y)$ and $g(x, y')$. For the specific case of *oblivious transfer* it is possible to define functions g_S and g_R (for the case of a corrupted sender and receiver, respectively) as follows:

- *Corrupted sender:* Since the sender is not supposed to learn anything about the receiver's input, we can just define $g_S((s_0, s_1), r) = r$. This then implies that a corrupted sender cannot distinguish the receiver's input bit r from a random bit.
- *Corrupted receiver:* In this case, the receiver is supposed to learn one of the sender's inputs, and nothing about the other. Thus, we define $g_R((s_0, s_1, r) = s_{1-r}$. This implies that a corrupted receiver cannot distinguish between the input bit of the sender that it did not receive and a random bit.

We now formally define the notion of semi-honest privacy with respect to a function g. We begin by presenting a general definition of this notion, and then present a specific instantiation for oblivious transfer.

Definition 2.3 *Let $\pi = (P_1, P_2)$ be a two-party protocol, let k denote the length of the parties' inputs, and let $g_1 : \{0,1\}^k \times \{0,1\}^k \mapsto \{0,1\}^*$ and $g_2 : \{0,1\}^k \times \{0,1\}^k \mapsto \{0,1\}^*$ be functions defined over the parties' inputs. We say that π is* semi-honest private with respect to a corrupted P_1 and function g_1*, if for every input $x \in \{0,1\}^k$ it holds that*

$$\{\mathsf{view}_{P_1}\langle P_1(1^n, x), P_2(1^n, U_k)\rangle, g_1(x, U_k)\} \stackrel{c}{\equiv} \left\{\mathsf{view}_{P_1}\langle P_1(1^n, x), P_2(1^n, U_k)\rangle, g_1(x, U'_k)\right\},$$

where U_k and U'_k are independent random variables. Likewise, π is semi-honest private with respect to a corrupted P_2 and function g_2*, if for every input $x \in \{0,1\}^k$*

$$\{\mathsf{view}_{P_2}\langle P_1(1^n, U_k), P_2(1^n, x)\rangle, g_2(U_k, x)\} \stackrel{c}{\equiv} \left\{\mathsf{view}_{P_2}\langle P_1(1^n, U_k), P_2(1^n, x)\rangle, g_2(U'_k, x)\right\}.$$

We say that π is semi-honest private with respect to (g_1, g_2) *if it is semi-honest private with respect to both g_1 and g_2.*

Observe that the definition requires that the joint distribution over P_1's input and the function g_1 be indistinguishable when g_1 is computed on both parties' real inputs x and U_k and when it is computed over P_1's real input x and an independent input U'_k for P_2. Observe also that the definition only requires privacy for a *random* input of the honest party; this suffices for our purposes. We now define semi-honest private oblivious transfer.

Definition 2.4 *A two-party protocol $\pi = (S, R)$ is a semi-honest private oblivious transfer if it is a non-trivial oblivious transfer protocol and it is semi-honest private with respect to (g_S, g_R), where g_S and g_R are as defined above.*

It is straightforward to verify that security against semi-honest parties, as defined in Section 2.2 (or in [13, Definition 7.2.1]) implies Definition 2.4.

Proposition 2.5 *Any protocol that computes the oblivious transfer functionality with security against semi-honest parties is a semi-honest private oblivious transfer protocol as in Definition 2.4.*

In fact, when considering oblivious transfer on strings of a constant size k (i.e., independent of the security parameter n) the two notions are equivalent. However, the notion of privacy with respect to a function will be more convenient for our purposes.

Privacy for defensible adversaries. As in the semi-honest case, we begin by presenting a general definition of privacy with respect to a function for defensible adversaries, and then present a specific instantiation of the definition for oblivious transfer. Observe that when defining privacy with respect to a function g for semi-honest adversaries, g is computed upon the corrupted party's input and either the honest party's real input or a random input. In the case of defensible adversaries, we wish to capture the intuition that an adversary should not be able to simultaneously present a good defense of its behavior and distinguish the case that g is computed upon the honest party's input or a random input. Now, unlike the case of semi-honest adversaries, a defensible adversary can use any input that it wishes in the protocol execution. This raises the question as to which input for the adversary is to be used when computing g. However, recall that an adversary's defense includes the input that it supposedly used. Thus, it is natural to use this input when computing g. Furthermore, this is exactly what we wish to require: if an adversary outputs a defense including an input x, then it should only be able to learn what it can learn when indeed using input x. Formally, we use the following definition.

Definition 2.6 *Let $\pi = (P_1, P_2)$ be a two-party protocol, let k denote the length of the parties' inputs, and let $g_1 : \{0,1\}^k \times \{0,1\}^k \mapsto \{0,1\}^*$ and $g_2 : \{0,1\}^k \times \{0,1\}^k \mapsto \{0,1\}^*$ be functions defined over the parties' inputs. We say that π is* defensibly private with respect to *(g_1, g_2), if for every family of polynomial-size circuits $\mathcal{A} = \{\mathcal{A}_n\}_{n\in\mathsf{N}}$*

1. *$\{\Gamma(v, g_1(x, U_k))\} \stackrel{c}{\equiv} \{\Gamma(v, g_1(x, U'_k))\}$, where $v = \mathsf{view}_{\mathcal{A}_n}\langle \mathcal{A}_n, P_2(1^n, U_k)\rangle$, $\Gamma(v, *)$ equals $(v, *)$ if following the execution $\mathcal{A}_n$ outputs a valid defense for π and $\perp$ otherwise, x is P_1's input in this defense, and U_k and U'_k are independent random variables.*

2. *$\{\Gamma(v, g_2(U_k, x))\} \stackrel{c}{\equiv} \{\Gamma(v, g_2(U'_k, x))\}$, where $v = \mathsf{view}_{\mathcal{A}_n}\langle P_1(1^n, U_k), \mathcal{A}_n\rangle$, and Γ, x, U_k and U'_k are as above.*

If only the first (resp. second) item holds, then we say that π is defensibly private with respect to a corrupted P_1 and function g_1 (resp., corrupted P_2 and g_2).

In order to see how this is applied to *oblivious transfer*, consider again the functions g_S and g_R above. Now, when the sender is corrupted (and so $\mathcal{A}$ plays the role of $S = P_1$) we have that whenever $\mathcal{A}$ outputs a good defense on a transcript t, it cannot distinguish the receiver's input bit

r from random. Furthermore, when the receiver is corrupted and $\mathcal{A}$ outputs a good defense on t that includes the bit r for input, then $\mathcal{A}$ cannot distinguish s_{1-r} from a random bit, where the inputs of S are (s_0, s_1).

Definition 2.7 *A two-party protocol $\pi = (S, R)$ is a defensibly private oblivious transfer if it is a non-trivial oblivious transfer protocol and it is defensibly private with respect to (g_S, g_R), where g_S and g_R are as defined above.*

2.5.1 An Alternative Formulation of Defensible Oblivious Transfer

For most of the reductions in this paper (with the exception of the first step from semi-honest to defensible adversaries), we find it easier to work with definitions of privacy for defensible adversaries that are based on "experiments". We present these definitions here and note that they are easily derived from Definition 2.6 when instantiated for oblivious transfer with functions g_S and g_R.

Privacy in the presence of a defensible sender. As we have seen, in an oblivious transfer protocol, the sender is not supposed to learn anything about the receiver's input. When considering a defensible adversary, this means that the adversary should not be able to simultaneously present a good defense of its behavior and obtain a significant advantage in guessing the value of the receiver's input. We stress that this privacy requirement only needs to hold when the sender outputs a good defense; in all other cases, there may be no privacy whatsoever. We begin by defining an adversarial experiment for a protocol π, an adversary $\mathcal{A}$ modeled by a (polynomial-size) family of circuits $\{\mathcal{A}_n\}_{n\in\mathsf{N}}$, and an input r for the receiver:

Experiment $\mathsf{Expt}^{\mathrm{snd}}_{\pi}(\mathcal{A}_n, r)$**:**

1. *Let ρ_R be a uniformly distributed random tape for the receiver R, and let $t = \langle \mathcal{A}_n, R(1^n, r, \rho_R)\rangle$. (That is, t is the transcript generated by an execution between the adversarial sender $\mathcal{A}_n$ and an honest R with input r and random tape ρ_R, running Protocol π.)*
2. *Let $((s_0, s_1, \rho_S), (\tau))$ be the output of $\mathcal{A}_n(t)$. (The triple (s_0, s_1, ρ_r) constitutes $\mathcal{A}_n$'s defense and τ is its guess for r.)*
3. *If (s_0, s_1, ρ_S) is a good defense by $\mathcal{A}_n$ for t in π, then output τ. Otherwise, output $\perp$.*

Given the above experiment, we can define privacy:

Definition 2.8 (privacy in the presence of a defensible sender): *Let $\pi = (S, R)$ be a non-trivial oblivious transfer protocol. We say that π is* private in the presence of a defensible sender *if for every family of polynomial-size circuits $\mathcal{A} = \{\mathcal{A}_n\}_{n\in\mathsf{N}}$ controlling S, every polynomial $p(\cdot)$ and for all sufficiently large n's*

$$\left|\Pr\left[\mathsf{Expt}^{\mathrm{snd}}_{\pi}(\mathcal{A}_n, 1) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathrm{snd}}_{\pi}(\mathcal{A}_n, 0) = 1\right]\right| < \frac{1}{p(n)}.$$

Privacy for random inputs in the presence of a defensible receiver. We now proceed to define privacy for defensible receivers. Recall that the receiver in an oblivious transfer protocol is supposed to obtain one of the pair (s_0, s_1) in the execution. However, the other value must remain secret. The fact that the receiver should learn something (unlike the sender), makes the definition a little more involved. As above, when considering defensible adversaries, the requirement is that, *as long as the adversary can provide a good defense,* it can only learn one of the values. Recall that by Definition 2.2, a party's defense includes its input (in this case, the bit r of the receiver, meaning that it wishes to obtain the value s_r). We therefore require that a defensible receiver can learn nothing about s_{1-r} when its defense contains the input value r. Due to technical reasons in our proofs later on, we define privacy only for the case that the sender's inputs are *uniformly distributed bits*. Fortunately, this will suffice for our constructions.

As above, we define an experiment for a protocol π and an adversary $\mathcal{A}$ modeled by a family of polynomial-size circuits $\{\mathcal{A}_n\}_{n\in\mathsf{N}}$. Informally, the experiment begins by choosing a random pair of bits (s_0, s_1) to be used for the sender's input. The adversary's aim is to guess the value of the input that it doesn't receive as output. Notice that unlike in $\mathsf{Expt}^{\mathrm{snd}}_{\pi}$, the sender's input is chosen as part of the experiment here.

Experiment $\mathsf{Expt}^{\mathrm{rec}}_{\pi}(\mathcal{A}_n)$**:**

1. *Choose* $s_0, s_1 \in_R \{0,1\}$ *uniformly at random.*
2. *Let* ρ_S *be a uniformly distributed random tape for* S *and let* $t = \langle S(1^n, s_0, s_1, \rho_S), \mathcal{A}_n\rangle$.
3. *Let* $((r, \rho_r), (\tau))$ *be the output of* $\mathcal{A}_n(t)$. *(The pair* (r, ρ_r) *constitute* $\mathcal{A}_n$*'s defense and* τ *is its guess for* s_{1-r}.*)*
4. *Output* 1 *if and only if* (r, ρ_r) *is a good defense by* $\mathcal{A}_n$ *for* t *in* π*, and* $\tau = s_{1-r}$.

Notice that by $\mathcal{A}$'s defense, it should have received s_r. The challenge of the adversary is therefore to guess the value of s_{1-r}; if it cannot do this, then the sender's privacy is preserved.

Definition 2.9 (privacy for random inputs in the presence of a defensible receiver): *Let* $\pi = (S, R)$ *be a non-trivial oblivious transfer protocol. We say that* π *is* private for random inputs in the presence of a defensible receiver *if for every family of polynomial-size circuits* $\mathcal{A} = \{\mathcal{A}_n\}_{n\in\mathsf{N}}$ *controlling* R*, for every polynomial* $p(\cdot)$ *and for all sufficiently large* n*'s*

$$\Pr\left[\mathsf{Expt}^{\mathrm{rec}}_{\pi}(\mathcal{A}_n) = 1\right] < \frac{1}{2} + \frac{1}{p(n)}$$

Remark: *Using the terminology of the first definition, we have that an oblivious transfer protocol is defensibly private if and only if it is private in the presence of a defensible sender and private for random inputs in the presence of a defensible receiver.*

2.5.2 Fully Secure Oblivious Transfer

All the above definitions refer only to "privacy", meaning that the adversary can learn nothing more about the honest party's input than what is revealed by the output. However, these definitions say nothing about the *simulatability* of the protocols in question. A protocol that is private by

one of the above definitions, or even private against unrestricted malicious parties, may not be secure against malicious parties according to the simulation-based definition from Section 2.2. For instance, the oblivious transfer protocols from [34, 1, 24] are private against malicious parties but they do not satisfy the stronger simulation-based definition (see Section 5.1 in [1] for discussion).

An *fully secure oblivious transfer* protocol (against malicious parties) is a protocol which securely computes the oblivious transfer functionality according to Definition 2.1. We note that in this case the definition can be simplified by not allowing the adversary to abort the ideal execution. It will be useful to separately consider security against a malicious sender and security against a malicious receiver. These notions can be naturally defined by restricting Definition 2.1 to the case $i = 1$ or $i = 2$. Since we will be constructing our protocols one step at a time, we will consider protocols that are private in the presence of defensible senders and secure in the presence of malicious senders, or vice versa.

3 From Semi-Honest Privacy to Defensible Privacy

In this section, we show how to transform (in a black-box way) any protocol that is semi-honest private with respect to a pair of functions (g_1, g_2) into a protocol that is defensibly private with respect to (g_1, g_2). The transformation shown here is generic and holds for all protocols, and thus the transformation for oblivious transfer is obtained as a special case. We say that two protocols π and π' have the same functionality if on any pair of inputs, the joint distributions of the outputs of honest parties running π and π' are identically distributed.

Theorem 3.1 *Let $\pi = (P_1, P_2)$ be a two-party protocol and let $g_1, g_2 : \{0,1\}^k \times \{0,1\}^k \mapsto \{0,1\}^*$ be functions defined over the parties' inputs. Assume that π is semi-honest private with respect to (g_1, g_2). Then there exists a fully black-box reduction from a protocol $\pi' = (P_1', P_2')$ to π and one-way functions, where π' and π have the same functionality and π' is defensibly private with respect to (g_1, g_2).*

The proof of Theorem 3.1 is obtained by applying the following lemma twice. Informally speaking, the lemma states that it is possible to "upgrade" the security of a protocol with respect to one of its parties while maintaining the initial security with respect to the other party. The lemma is applied in different directions, once in order to upgrade the security of the first party and another time to upgrade the security of the second party. In order to make this clear, we write the transformation for a protocol $\pi = (A, B)$ with parties A and B. Then in one application of the lemma A is set to P_1 and B is set to P_2, and in the other application of the lemma A is set to P_2 and B is set to P_1.

Lemma 3.2 *Let $\pi = (A, B)$ be a two-party protocol and let $g_A, g_B : \{0,1\}^k \times \{0,1\}^k \mapsto \{0,1\}^*$ be functions defined over the parties' inputs. Assume that π is semi-honest private with respect to a corrupted A and g_A. Then, there exists a fully black box reduction from a protocol $\pi' = (A', B')$ to π and one-way functions, where π' and π have the same functionality and π' is defensibly private with respect to a corrupted A' and g_A. Furthermore, if π is semi-honest (resp., defensibly) private with respect to a corrupted B and g_B then π' is semi-honest (resp., defensibly) private with respect to a corrupted B' and g_B.*

Proof: We define a protocol π' that uses π and a commitment scheme Com in a black-box way. This suffices, because there exists a fully black-box reduction from Com to one-way functions

(see Theorem A.3). In the new protocol A' first commit (using Com) to an input string x_A and a random string $\rho_{A'}$, then B' sends a random string $\rho_{B'}$ to A', and finally the two parties engage in the protocol π, where A' uses x_A and $\rho_{A'} \oplus \rho_{B'}$ as the input and random coins of A respectively.

The hiding guarantee of Com yields that π' is as "safe" for A' as π is for A. Where the binding property of Com yields that when giving a valid defense (which in particular contains a valid decommitment of Com), A' acts in the embedded π as the honest party A would on a *predetermined* input. Namely, π' is defensibly private with respect to (a corrupted) A'.

In the following we formally define π' and then prove its security.

Protocol 3.3 (The defensible protocol $\pi' = (A', B')$)

Common input: 1^n

A''s input and random-tape: $x_A \in \{0,1\}^k$ *and* $\rho_{A'} = (\rho^1_{A'}, \rho^2_{A'})$

B''s input and random-tape: $x_B \in \{0,1\}^k$ *and* $\rho_{B'} = (\rho^1_{B'}, \rho^2_{B'}, \rho^3_{B'})$

1. *A' and B' run an execution of Com in which A' commits to $(x_A, \rho^2_{A'})$; the parties use security parameter 1^n, and random coins $\rho^1_{A'}$ and $\rho^1_{B'}$, respectively.*
2. *B' sends $\rho^3_{B'}$ to A'.*
3. *A' and B' run the protocol $\pi = (A, B)$, where A' runs A with security parameter 1^n, input x_A and random-tape $\rho^2_{A'} \oplus \rho^3_{B'}$, and B' runs B with security parameter 1^n, input x_B and random-tape $\rho^2_{B'}$.*
4. *A' outputs whatever A outputs in π and B' outputs whatever B outputs in π.*

It is immediate that π' has the same functionality as π. It remains to show that π' maintains the *same* level of privacy with respect to a corrupted B' and g_B, and that π' is defensibly private with respect to A' and g_A. The fact that the same level of privacy is kept with respect to B' is shown in Lemma 3.4 below, and the fact that defensible privacy is obtained with respect to a corrupted A' is shown in Lemma 3.5; this latter lemma is the heart of the proof.

Lemma 3.4 *If π is semi-honest (resp., defensibly) private with respect to a corrupted B and function g_B then π' is semi-honest (resp., defensibly) private with respect to a corrupted B' and g_B.*

Proof: We assume that π is semi-honest private with respect to a corrupted B and g_B and prove that π' is also semi-honest private with respect to a corrupted B' and g_B; the proof for the defensibly private case is analogous. First, observe that the messages that B' receives in π', and thus its view, consist of a commitment from A' (to x_A and $\rho^2_{A'}$) followed by an execution of π. Thus, we can write the view of B' in π' as follows:

$$\begin{aligned}\mathsf{view}_{B'}\langle A'(1^n, x_A, \rho^1_{A'}, \rho^2_{A'}), B'(1^n, x_B, \rho^1_{B'}, \rho^2_{B'}, \rho^3_{B'})\rangle \\ = (\mathsf{Com}(x_A, \rho^2_{A'}), \mathsf{view}_B\langle A(1^n, x_A, \rho^2_{A'} \oplus \rho^3_{B'}), B(1^n, x_B, \rho^2_{B'})\rangle. \quad (1)\end{aligned}$$

Now, by a straightforward reduction to the hiding property of Com, we have that for every $x_B \in \{0,1\}^k$

$$\begin{aligned}\big\{(\mathsf{Com}(U_k, \rho^2_{A'}), \mathsf{view}_B\langle A(1^n, U_k, \rho^2_{A'} \oplus \rho^3_{B'}), B(1^n, x_B, \rho^2_{B'})\rangle, g_B(U_k, x_B)\big\} \\ \stackrel{c}{\equiv} \Big\{(\mathsf{Com}(0^\ell), \mathsf{view}_B\langle A(1^n, U_k, \rho^2_{A'} \oplus \rho^3_{B'}), B(1^n, x_B, \rho^2_{B'})\rangle, g_B(U_k, x_B)\Big\}\end{aligned}$$

where ℓ is the combined length of U_k and $\rho^2_{A'}$ (note that the input x_A of A' is taken to be a random value U_k here). Next, since π is semi-honest private with respect to a corrupted B and function g_B, we have that

$$\left\{(\mathsf{Com}(0^\ell), \mathsf{view}_B\langle A(1^n, U_k, \rho^2_{A'} \oplus \rho^3_{B'}), B(1^n, x_B, \rho^2_{B'})\rangle, g_B(U_k, x_B)\right\}$$
$$\stackrel{c}{\equiv} \left\{(\mathsf{Com}(0^\ell), \mathsf{view}_B\langle A(1^n, U_k, \rho^2_{A'} \oplus \rho^3_{B'}), B(1^n, x_B, \rho^2_{B'})\rangle, g_B(U'_k, x_B)\right\}$$

Applying the hiding property of the commitment scheme once again, we have that

$$\left\{(\mathsf{Com}(0^\ell), \mathsf{view}_B\langle A(1^n, U_k, \rho^2_{A'} \oplus \rho^3_{B'}), B(1^n, x_B, \rho^2_{B'})\rangle, g_B(U'_k, x_B)\right\}$$
$$\stackrel{c}{\equiv} \left\{(\mathsf{Com}(U_k, \rho^2_{A'}), \mathsf{view}_B\langle A(1^n, U_k, \rho^2_{A'} \oplus \rho^3_{B'}), B(1^n, x_B, \rho^2_{B'})\rangle, g_B(U'_k, x_B)\right\}$$

Combining the above three equations we have that

$$\left\{(\mathsf{Com}(U_k, \rho^2_{A'}), \mathsf{view}_B\langle A(1^n, U_k, \rho^2_{A'} \oplus \rho^3_{B'}), B(1^n, x_B, \rho^2_{B'})\rangle, g_B(U_k, x_B)\right\}$$
$$\stackrel{c}{\equiv} \left\{(\mathsf{Com}(U_k, \rho^2_{A'}), \mathsf{view}_B\langle A(1^n, U_k, \rho^2_{A'} \oplus \rho^3_{B'}), B(1^n, x_B, \rho^2_{B'})\rangle, g_B(U'_k, x_B)\right\}.$$

However, by Eq. (1), this is equivalent to writing that

$$\left\{\mathsf{view}_{B'}\langle A'(1^n, U_k, \rho^1_{A'}, \rho^2_{A'}), B'(1^n, x_B, \rho^1_{B'}, \rho^2_{B'}, \rho^3_{B'})\rangle, g_B(U_k, x_B)\right\}$$
$$\stackrel{c}{\equiv} \left\{\mathsf{view}_{B'}\langle A'(1^n, U_k, \rho^1_{A'}, \rho^2_{A'}), B'(1^n, x_B, \rho^1_{B'}, \rho^2_{B'}, \rho^3_{B'})\rangle, g_B(U'_k, x_B)\right\}.$$

Thus, π' is semi-honest private for a corrupted B' and function g_B, as in Definition 2.3. ■

We now prove the more interesting part of the lemma; the fact that π' is defensibly private with respect to a corrupted A'.

Lemma 3.5 *If π is semi-honest private with respect to a corrupted A and function g_A, then π' is defensibly private with respect to A' and g_A.*

Proof: Assume by contradiction that there exists a probabilistic polynomial-time adversary $\mathcal{A}'$ and a distinguisher D' that violate the defensible privacy of π with respect to A' and g_A. Hence, we assume without loss of generality that there exists a polynomial $p(\cdot)$ be such that for infinitely many n's:

$$\Pr[D(\Gamma(v, g_1(x, U_k))) = 1] - D(\Gamma(v, g_1(x, U'_k))) = 1] \geq \frac{1}{p(n)}, \tag{2}$$

where v is $\mathcal{A}'_n$'s view in $\langle \mathcal{A}'_n, B(1^n, U_k)\rangle$, $\Gamma(v, *)$ equals $(v, *)$ if following the execution $\mathcal{A}_n$ outputs a valid defense for π and $\perp$ otherwise, x is A's input in this defense, and U_k and U'_k are independent random variables. We now use $\mathcal{A}'$ and D' to construct an efficient distinguisher D with oracle access to $\mathcal{A}'$ and D' that violates the semi-honest privacy of (A, B) with respect to A and g_A. Recall that in order to violate the semi-honest privacy of (A, B), algorithm D first picks an input element x_A (this is implicit in the fact that in Definition 2.3 we quantify over all inputs). Then, D receives A's view from the execution of $\langle A(1^n, x_A), B(1^n, U_k)\rangle$ together with a value y that equals either $g_A(x_A, U_k)$ or $g_A(x_A, U'_k)$, and it attempts to distinguish between these two. In order to make the dependencies between the two stages explicit, D uses the variable z to transfer information from the input sampling stage to the distinguishing stage.

Algorithm 3.6 (The distinguisher D)

Sampling stage:

- **Input:** 1^n
- **Instructions:**
  1. *Choose uniformly distributed random values for $\rho_{\mathcal{A}'}$ and $\rho^1_{B'}$, and fix $\mathcal{A}'$'s random coins to $\rho_{\mathcal{A}'}$.*
  2. *Simulate the first step of $(\mathcal{A}', B')$ (i.e., the execution of Com), where B' uses $\rho^1_{B'}$ as its random coins.*
  3. *Run the following $n \cdot p(n)$ times:*
     (a) *Simulate the last two steps of $(\mathcal{A}', B')$, choosing B''s input and random coins (i.e., x_B, $\rho^2_{B'}$ and $\rho^3_{B'}$) uniformly at random.*
     (b) *If $\mathcal{A}'$ outputs a valid defense d, then set $x_A = x_{\mathcal{A}'}$ and $z = (\rho_{\mathcal{A}'}, \rho^1_{B'}, \rho^2_{A'})$, where $x_{\mathcal{A}'}$ and $\rho^2_{A'}$ are the values of these input variables in d, and return.*[4]
  4. *If no valid defense is obtained, then set $z = \perp$ and an arbitrary value for x_A.*

Predicting stage:

- **Input:** *a tuple (z, v_A, c) where z equals either $(\rho_{\mathcal{A}'}, \rho^1_{B'}, \rho^2_{A'})$ or $\perp$, v_A is A's view $\mathsf{view}_A\langle A(1^n, x_A), B(1^n, U_k)\rangle$ from the semi-honest execution with x_A from the sampling stage, and c is either $g_A(x_A, U_k)$ or $g_A(x_A, U'_k)$.*
- **Instructions:**
  1. *If $z = \perp$, output a random coin and return.*
  2. *Fix the random coins of $\mathcal{A}'$ to $z[\rho_{\mathcal{A}'}]$, where $z[\rho_{\mathcal{A}'}]$ denotes the first element $\rho_{\mathcal{A}'}$ in the tuple z.*
  3. *Simulate the first step of $(\mathcal{A}', B')$ (i.e., the execution of Com), using random coins $z[\rho^1_{B'}]$ (the second value in the tuple z) for B''s random coins.*
  4. *Simulate the second step of $(\mathcal{A}', B')$, by simulating B' sending $\rho^3_{B'} = v_A[\rho_A] \oplus z[\rho^2_{A'}]$ to $\mathcal{A}'$, where $v_A[\rho_A]$ are A's coins in the view v_A and $z[\rho^2_{A'}]$ is the third element of z. (Observe that this implies that $z[\rho^2_{A'}] \oplus \rho^3_{B'}$ equals the random-tape ρ_A of the semi-honest A.)*
  5. *Simulate the last step of $(\mathcal{A}', B')$, where B' sends the same messages that B sends in v_A (recall that v_A is A's view in the semi-honest execution of π).*
  6. *Let $v_{\mathcal{A}'}$ be the view of $\mathcal{A}'$ at the end of the above simulation.*
     If $\mathcal{A}'$ outputs a good defense for $v_{\mathcal{A}'}$, then D outputs what D' outputs when given input $(v_{\mathcal{A}'}, c)$. Otherwise, D outputs a random coin.

It is easy to verify that D is efficient given oracle access to $\mathcal{A}'$ and D'. We now show that D violates the semi-honest privacy of π with respect to A and g_A.

We consider a random execution of (D, A, B) with security parameter 1^n, respective inputs x_A and x_B, where x_A is chosen in the sampling phase of D and x_B is chosen at random, and respective

[4] Observe that when $\mathcal{A}'$ provides a valid defense, it decommits to the committed pair $(x_A, \rho^2_{A'})$. Furthermore, if $\mathcal{A}'$ is able to provide a valid defense then it must use random coins $\rho_A = \rho^2_{A'} \oplus \rho^3_{B'}$ where $\rho^3_{B'}$ is the string it received in step 1 of the protocol.

random tapes ρ_A and ρ_B. Let $z = (\rho_{\mathcal{A}'}, \rho^1_{B'}, \rho^2_{A'})$ be the output of D from the sampling stage, let trans be the transcript of the simulation of π' carried out by D in the predicting stage (where $\mathsf{trans} = \perp$ if $z = \perp$). Finally, define the random variable Sim (which is a simulation of π' based on π) by

$$\text{Sim} = \left\{(x_A, x_B, z[\rho_{\mathcal{A}'}], \rho_{B'} = (z[\rho^1_{B'}], \rho_B, \rho_A \oplus z[\rho^2_{\mathcal{A}'}]), \mathsf{trans})\right\}$$

where all elements are distributed as above, and if $z = \perp$, then all the variables that depend on z are set to $\perp$. Intuitively, Sim represents a simulation of an execution of π' based on π. Note that the random-tape of the defensible $\mathcal{A}'$ is $\rho_{\mathcal{A}'}$ (as output in z by D) and the random-tape of B' is set so that the third element (which is sent to A' in step 2 of protocol π') equals $\rho^3_{B'} = \rho_A \oplus z[\rho^2_{\mathcal{A}'}]$.

This essentially[5] implies that if $\mathcal{A}'$ is able to provide a defense, then it must use random coins $\rho^3_{B'} \oplus \rho^2_{A'} = \rho_A$ in the step in which the semi-honest protocol π is run. Recall that ρ_A is the random-tape of A in the real semi-honest execution of π.

Let $\mathsf{Supp}(X)$ denote the support of the random variable X. Given $s \in \mathsf{Supp}(\text{Sim})$, let $\mathsf{defense}(s)$ denote the defense that $\mathcal{A}'$ outputs upon random-tape and execution transcript $s[\rho_{\mathcal{A}'}]$ and $s[\mathsf{trans}]$ respectively, and let $\mathsf{IsGoodDef}(s)$ be a predicate that equals 1 if and only if $\mathsf{defense}(s)$ is a *good defense*. For $s \in \mathsf{Supp}(\text{Sim})$ and $c \in Im(g_A)$, let $\mathsf{Out}_D(s, c)$ be the output bit of D induced by s and c — $\mathcal{A}'$ is simulated on random-tape and execution transcript $s[\rho_{\mathcal{A}'}]$ and $s[\mathsf{trans}]$, and then D' is executed on $(v_{\mathcal{A}'}, c)$. Observe that $\mathsf{Out}_D(s, c)$ is a random variable that depends only on the random coins used to invoke D', and that for any value of c this is the same random variable as the output of D (on the same c) in any execution of (D, A, B) that "yields" s. Finally, let $\mathsf{Adv}_D(s)$ be the advantage of D in distinguishing between the case that c is obtained by computing g_A using B's real input and a random input, given s. That is,

$$\mathsf{Adv}_D(s) = \Pr[\mathsf{Out}_D(s, g_A(s[x_A], s[x_B])) = 1] - \Pr[\mathsf{Out}_D(s, g_A(s[x_A], U_k)) = 1]$$

Since Sim is a deterministic function of the random variable that describes the execution of (D, A, B), it follows that $\mathsf{Exp}[\mathsf{Adv}_D(\text{Sim})]$ is exactly the advantage of D in breaking the semi-honest privacy of π with respect to A and g_A.

We would like to relate the above success probability to that of D distinguishing, after an execution of π', between the case that c is obtained by computing g_A using B's real input and using a random input. Let Real be the random variable Real $= (x_A, x_B, \rho_{\mathcal{A}'}, \rho_{B'}, \mathsf{trans})$ induced by an execution of $(\mathcal{A}', B')$ with security parameter 1^n, where x_A is the input of A' in the defense of $\mathcal{A}'$ (we set $x_A = \perp$ if no good defense is given). Let $\mathsf{Out}_{D'}(s, c)$ be the output bit of D' given s and c if $\mathsf{IsGoodDef}(s) = 1$, and a random coin if otherwise, and let $\mathsf{Adv}_{D'}(s)$ be the advantage of D' in distinguishing the case that c is obtained by computing g_A using B's real input and a random input, given s. As above, it is easy to verify that $|\mathsf{Exp}[\mathsf{Adv}_{D'}(\text{Real})]|$ is exactly the advantage of D' in breaking the defensible privacy of π' with respect to A' and g_A.

We use the following claim to relate the advantage of D in breaking the semi-honest privacy of π to that of D' in breaking the defensible privacy of π'.

Claim 3.7 *For all large enough n, there exists a set $L \subseteq \mathsf{Supp}(\text{Real})$ for which the following hold:*

1. *The event $\{\mathsf{IsGoodDef}(s) \wedge s \notin L\}$ happens with probability at most $\frac{1}{4p(n)}$, for s that is sampled according to either* Real *or* Sim*,*

[5] Unless $\mathcal{A}'$ decommits in the predicting stage to different value than the one it decommitted to in the sampling stage, but this can only happen with negligible probability.

2. $(1-\frac{1}{4p(n)})\cdot \Pr[\text{Real}=s] \leq \Pr[\text{Sim}=s] \leq \Pr[\text{Real}=s]$, *for every* $s \in L$,

where $p(\cdot)$ *is from Eq. (2).*

Informally, L contains all the transcripts for which $\mathcal{A}'$ has a significant chance to provide a good defense, both with respect to Real and with respect to Sim, where in addition, Real and Sim induces essentially the same distribution on L. It follows that the advantage of $\mathcal{A}'$ in predicting g_1 in π' while providing a good defense comes from transcript inside L, and that D is most likely to flip a random coin as output when outside L. Hence, the advantage of D in predicting g_1 in π is noticeable. Before proving Claim 3.7, let us first follows the above intuition for formally proving Lemma 3.5.

Let n be large enough so that Claim 3.7 holds. Eq. (2) yields that

$$\begin{aligned}
&\mathsf{Exp}[\mathsf{Adv}_{D'}(\text{Real})\cdot\chi_L(\text{Real})] \qquad (3)\\
&= \mathsf{Exp}[\mathsf{Adv}_{D'}(\text{Real})] - \mathsf{Exp}[\mathsf{Adv}_{D'}(\text{Real})\cdot\overline{\chi_L(\text{Real})}]\\
&= \mathsf{Exp}[\mathsf{Adv}_{D'}(\text{Real})] - \mathsf{Exp}[\mathsf{Adv}_{D'}(\text{Real})\cdot\mathsf{IsGoodDef}(\text{Real})\cdot\overline{\chi_L(\text{Real})}]\\
&\geq \frac{1}{p(n)} - \Pr[\mathsf{IsGoodDef}(\text{Real})\wedge \text{Real}\notin L] \geq \frac{1}{p(n)} - \frac{1}{4p(n)},
\end{aligned}$$

where $\chi_L(s)$ is one if $s \in L$ and zero otherwise. The second equality holds since $\mathsf{Adv}_{D'}(s)=0$ for s with $\mathsf{IsGoodDef}(s)=0$, and the last inequality is by Claim 3.7(1). Similarly, Claim 3.7(1) yields that

$$\begin{aligned}
&\mathsf{Exp}\left[\mathsf{Adv}_D(\text{Sim})\right] \qquad (4)\\
&= \mathsf{Exp}\left[\mathsf{Adv}_D(\text{Sim})\cdot\chi_L(\text{Sim})\right] + \mathsf{Exp}\left[\mathsf{Adv}_D(\text{Sim})\cdot\overline{\chi_L(\text{Sim})}\right]\\
&= \mathsf{Exp}\left[\mathsf{Adv}_D(\text{Sim})\cdot\chi_L(\text{Sim})\right] + \mathsf{Exp}\left[\mathsf{Adv}_D(\text{Sim})\cdot\overline{\chi_L(\text{Sim})}\cdot\mathsf{IsGoodDef}(\text{Sim})\right]\\
&\geq \mathsf{Exp}\left[\mathsf{Adv}_D(\text{Sim})\cdot\chi_L(\text{Sim})\right] - \Pr\left[\text{Sim}\notin L\wedge\mathsf{IsGoodDef}(\text{Sim})\right]\\
&\geq \mathsf{Exp}\left[\mathsf{Adv}_D(\text{Sim})\cdot\chi_L(\text{Sim})\right] - \frac{1}{4p(n)},
\end{aligned}$$

where the second equality holds since $\mathsf{Adv}_D(s)=0$ for s with $\mathsf{IsGoodDef}(s)=0$. Since $\mathsf{Out}_D(s,c)$ and $\mathsf{Out}_{D'}(s,c)$ are distributed the same, Claim 3.7(2) yields that

$$\begin{aligned}
&\Pr\left[\mathsf{Out}_D(\text{Sim}, g_A(\text{Sim}[x_A], U_k)) = 1\wedge \text{Sim}\in L\right] \qquad (5)\\
&\quad= \sum_{s\in L}\Pr[\text{Sim}=s]\cdot\Pr\left[\mathsf{Out}_{D'}(s, g_A(s[x_{\mathcal{A}'}], U_k)) = 1\right]\\
&\quad\leq \sum_{s\in L}\Pr[\text{Real}=s]\cdot\Pr\left[\mathsf{Out}_{D'}(s, g_A(s[x_{\mathcal{A}'}], U_k)) = 1\right]\\
&\quad= \Pr\left[\mathsf{Out}_{D'}(\text{Real}, g_A(\text{Real}[x_{\mathcal{A}'}], U_k)) = 1\wedge \text{Real}\in L\right],
\end{aligned}$$

and similarly,

$$\begin{aligned}
&\Pr\left[\mathsf{Out}_D(\text{Sim}, g_A(\text{Sim}[x_A, x_B])) = 1\wedge \text{Sim}\in L\right] \qquad (6)\\
&\quad\geq \left(1-\frac{1}{4p(n)}\right)\cdot\Pr\left[\mathsf{Out}_{D'}(\text{Real}, g_A(\text{Real}[x_{\mathcal{A}'}, x_B])) = 1\wedge \text{Real}\in L\right].
\end{aligned}$$

Eq. (4) and the definition of Adv_D yield that

$$\begin{aligned}
&\mathsf{Exp}[\mathsf{Adv}_D(\mathrm{Sim})] \\
&\geq \Pr\left[\mathsf{Out}_D(\mathrm{Sim}, g_A(\mathrm{Sim}[x_A, x_B])) = 1 \wedge \mathrm{Sim} \in L\right] \\
&\quad - \Pr\left[\mathsf{Out}_D(\mathrm{Sim}, g_A(\mathrm{Sim}[x_A], U_k)) = 1 \wedge \mathrm{Sim} \in L\right] - \frac{1}{4p(n)}.
\end{aligned}$$

We conclude that

$$\begin{aligned}
&\mathsf{Exp}[\mathsf{Adv}_D(\mathrm{Sim})] \\
&\geq \left(1 - \frac{1}{4p(n)}\right) \cdot \Pr\left[\mathsf{Out}_{D'}(\mathrm{Real}, g_A(\mathrm{Real}[x_{\mathcal{A}'}, x_B])) = 1 \wedge \mathrm{Real} \in L\right] \\
&\quad - \Pr\left[\mathsf{Out}_{D'}(\mathrm{Real}, g_A(\mathrm{Real}[x_{\mathcal{A}'}], U_k)) = 1 \wedge \mathrm{Real} \in L\right] - \frac{1}{4p(n)} \\
&\geq \left(1 - \frac{1}{4p(n)}\right) \cdot (\Pr\left[\mathsf{Out}_{D'}(\mathrm{Real}, g_A(\mathrm{Real}[x_{\mathcal{A}'}, x_B])) = 1 \wedge \mathrm{Real} \in L\right] \\
&\quad - \Pr\left[\mathsf{Out}_{D'}(\mathrm{Real}, g_A(\mathrm{Real}[x_{\mathcal{A}'}], U_k)) = 1 \wedge \mathrm{Real} \in L\right]) - \frac{2}{4p(n)} \\
&= \left(1 - \frac{1}{4p(n)}\right) \cdot \mathsf{Exp}\left[\mathsf{Adv}_{D'}(\mathrm{Real}) \cdot \chi_L(\mathrm{Real})\right] - \frac{2}{4p(n)} \\
&\geq \left(1 - \frac{1}{4p(n)}\right) \cdot \left(\frac{1}{p(n)} - \frac{1}{4p(n)}\right) - \frac{1}{2p(n)} \\
&> \frac{1}{8p(n)},
\end{aligned}$$

where the first inequality follows by Eq. (5) and Eq. (6), the third one by Eq. (3), and for the last inequality we assume without loss of generality that $p(n) > 8$. Since the above holds for infinitely many n's, this concludes the proof of Lemma 3.5 and thus the proof of Lemma 3.2.

■

We turn to prove Claim 3.7.

Proof: For $s \in \mathsf{Supp}(\mathrm{Real})$, let $\mathsf{decom}(s)$ equal the decommitment of Com that appears in $\mathsf{defense}(s)$ when s is a good defense, and let it equal $\perp$ otherwise (note that since s contains all random tapes, this value is uniquely determined). Now, for any string $\sigma \in \{0,1\}^*$ and pair of given random tapes $(\rho^1_{B'}, \rho_{\mathcal{A}'})$, we denote

$$\mathsf{DecomPr}_\sigma(\rho^1_{B'}, \rho_{\mathcal{A}'}) \stackrel{\text{def}}{=} \Pr\left[\mathsf{decom}(\mathrm{Real}) = \sigma \mid \mathrm{Real}[\rho^1_{B'}, \rho_{\mathcal{A}'}] = (\rho^1_{B'}, \rho_{\mathcal{A}'})\right].^{6}$$

Observe that as long as the binding property holds with respect to the commitment scheme, this means that the decommitment value of $\mathrm{Real}[(\rho^1_{B'}, \rho_{\mathcal{A}'})]$ is *unique*. Stated differently, if there exist multiple values σ such that $\mathsf{DecomPr}_\sigma(\rho^1_{B'}, \rho_{\mathcal{A}'})$ is non-negligible, then this implies the existence of an adversary who breaks the binding of the commitment scheme; we describe this adversary in more detail below.

[6] We define $\mathsf{DecomPr}$ as a function of $\rho^1_{B'}$ and $\rho_{\mathcal{A}'}$ because the messages sent in the commitment phase depend only on $\mathcal{A}'$'s random-tape $\rho_{\mathcal{A}'}$ and the random coins $\rho^1_{B'}$ used by B' in the commitment phase.

For any $(\rho^1_{B'}, \rho_{\mathcal{A}'})$, let $\mathsf{Heaviest}(\rho^1_{B'}, \rho_{\mathcal{A}'})$ be the most likely decommitment given that these values were used in the commitment stage. Namely,

$$\mathsf{Heaviest}(\rho^1_{B'}, \rho_{\mathcal{A}'}) = \mathsf{argmax}_{\sigma\in\{0,1\}^*} \left(\mathsf{DecomPr}_\sigma(\rho^1_{B'}, \rho_{\mathcal{A}'})\right),$$

and let $\neg\mathsf{Heaviest}(\rho^1_{B'}, \rho_{\mathcal{A}'}) = \{0,1\}^t \setminus \{\mathsf{Heaviest}(\rho^1_{B'}, \rho_{\mathcal{A}'})\}$, where t is the length of $(x_A, \rho^2_{A'})$ (i.e., $\neg\mathsf{Heaviest}(\rho^1_{B'}, \rho_{\mathcal{A}'})$ contains all possible decommitment values excluding the most likely one). For a finite set $S \subseteq \{0,1\}^*$, we let $\mathsf{DecomPr}_S(\rho^1_{B'}, \rho_{\mathcal{A}'}) \stackrel{\text{def}}{=} \sum_{\sigma\in S} \mathsf{DecomPr}_\sigma(\rho^1_{B'}, \rho_{\mathcal{A}'})$, and define the set L as follows:

$$L = \left\{ s \in \mathsf{Supp}(\mathrm{Real}) \;\middle|\; \begin{array}{l} \mathsf{decom}(s) = \mathsf{Heaviest}(s[\rho^1_{B'}, \rho_{\mathcal{A}'}]) \;\wedge \\ \mathsf{DecomPr}_{\mathsf{Heaviest}}(s[\rho^1_{B'}, \rho_{\mathcal{A}'}]) \geq \frac{1}{8p(n)} \;\wedge \\ \mathsf{DecomPr}_{\neg\mathsf{Heaviest}}(s[\rho^1_{B'}, \rho_{\mathcal{A}'}]) \leq \frac{1}{8np(n)^2} \end{array} \right\}.$$

Note that each $s \in L$ contains a valid decommitment, whose value can be extracted efficiently given only the prefix of s (i.e., $s[\rho^1_{B'}, \rho_{\mathcal{A}'}]$) by sampling from Real conditioned on this prefix (as done in the sampling stage of D). Moreover, since $\mathsf{DecomPr}_{\neg\mathsf{Heaviest}}$ is small, it is unlikely that this prefix yields a different decommitment. Hence, if the prefix of s is sampled in the sampling stage of D, then with very high probability D extracts the decommitment of s in this stage. Moreover, conditioned on the event that D was successful in the sampling stage, it is easy to verify that the probability of s with respect to Sim and with respect to Real are essentially the same, which establishes the second property of L stated by Claim 3.7. The first property follows since it is unlikely that $\mathcal{A}'$, or D, generates a transcript $s \notin L$ with $\mathsf{IsGoodDef}(s) = 1$ — the binding property of the commitment yields that it is unlikely for s such that $\mathsf{decom}(s) \neq \mathsf{Heaviest}(s[\rho^1_{B'}, \rho_{\mathcal{A}'}])$, and by Markov inequality it is also unlikely for s such that $\mathsf{decom}(s) = \mathsf{Heaviest}(s[\rho^1_{B'}, \rho_{\mathcal{A}'}])$. In the following we formalize the above observations for proving the two properties of L required by Claim 3.7.

Proving 1. We start by upper bounding the probability of the event $\{\mathsf{IsGoodDef}(\mathrm{Real}) \wedge \mathrm{Real} \notin L\}$. Recall this even happens if one of the following events holds:

1. $\{\mathsf{IsGoodDef}(\mathrm{Real}) \wedge \mathsf{decom}(\mathrm{Real}) \neq \mathsf{Heaviest}(\mathrm{Real}[\rho^1_{B'}, \rho_{\mathcal{A}'}])\}$
2. $\{\mathsf{IsGoodDef}(\mathrm{Real}) \wedge \mathsf{DecomPr}_{\mathsf{Heaviest}}(\mathrm{Real}[\rho^1_{B'}, \rho_{\mathcal{A}'}]) < \frac{1}{8p(n)}\}$
3. $\{\mathsf{IsGoodDef}(\mathrm{Real}) \wedge \mathsf{DecomPr}_{\neg\mathsf{Heaviest}}(\mathrm{Real}[\rho^1_{B'}, \rho_{\mathcal{A}'}]) > \frac{1}{8np(n)^2}\}$

For the sake of clarity in our analysis below, we use the shorthand decom to denote the first event, $\mathsf{DecomPr}_{\mathsf{Heaviest}}$ to denote the second event, and $\mathsf{DecomPr}_{\neg\mathsf{Heaviest}}$ to denote the third event. Using this notation, we have:

$$\begin{aligned} &\Pr\left[\mathsf{IsGoodDef}(\mathrm{Real}) \wedge \mathrm{Real} \notin L\right] \\ &= \Pr[\mathsf{DecomPr}_{\neg\mathsf{Heaviest}}] + \Pr[\mathsf{DecomPr}_{\mathsf{Heaviest}} \wedge \neg\mathsf{DecomPr}_{\neg\mathsf{Heaviest}}] \\ &\quad + \Pr[\mathsf{decom} \wedge \neg\mathsf{DecomPr}_{\mathsf{Heaviest}} \wedge \neg\mathsf{DecomPr}_{\neg\mathsf{Heaviest}}] \\ &\leq \Pr[\mathsf{DecomPr}_{\neg\mathsf{Heaviest}}] + \Pr[\mathsf{DecomPr}_{\mathsf{Heaviest}} \wedge \neg\mathsf{DecomPr}_{\neg\mathsf{Heaviest}}] + \Pr[\mathsf{decom} \wedge \neg\mathsf{DecomPr}_{\neg\mathsf{Heaviest}}]. \end{aligned}$$

We proceed to bound each of these three terms. For bounding the first term, we observe that for every polynomial $q(\cdot)$ and sufficiently large n,

$$\Pr\left[\mathsf{DecomPr}_{\neg\mathsf{Heaviest}}(\mathrm{Real}[\rho^1_{B'}, \rho_{\mathcal{A}'}]) > \frac{1}{q(n)}\right] < \frac{1}{q(n)}. \tag{7}$$

If this does not hold, then there exists a value $\sigma' \neq \sigma \stackrel{\text{def}}{=} \mathsf{Heaviest}(\text{Real}[\rho^1_{B'}, \rho_{\mathcal{A}'}])$ such that

$$\mathsf{DecomPr}_{\sigma'}(\text{Real}[\rho^1_{B'}, \rho_{\mathcal{A}'}]) \geq 1/q(n).$$

This in turn implies that

$$\mathsf{DecomPr}_{\sigma}(\text{Real}[\rho^1_{B'}, \rho_{\mathcal{A}'}]) > 1/q(n),$$

because σ is at least as "heavy" as σ' (recall the definition of $\mathsf{Heaviest}(\cdot)$). We conclude that there are two distinct values σ and σ' that are decommitted to with non-negligible probability. As we have discussed above, this implies the existence of an adversary $\mathcal{A}_{\mathsf{Com}}$ that violates the binding property of Com. Formally, adversary $\mathcal{A}_{\mathsf{Com}}$ runs $\mathcal{A}'$ in the first step of Protocol π' using random coins $\rho_{\mathcal{A}'}$ for $\mathcal{A}'$ and coins $\rho^1_{B'}$ for B'. Next, it simulates the rest of the protocol twice (with the same prefix) and outputs the two decommitments implied by $\mathcal{A}'$'s defenses, if there are two. By what we have seen, if $\mathsf{DecomPr}_{\neg\mathsf{Heaviest}}(\text{Real}[\rho^1_{B'}, \rho_{\mathcal{A}'}]) > \frac{1}{q(n)}$ holds then $\mathcal{A}_{\mathsf{Com}}$ obtains decommitments to two different values σ and σ' with probability $1/q^2(n)$ ($1/q(n)$ independently for each). Thus, if the contradicting assumption holds and this occurs with probability at least $1/q(n)$, we have that $\mathcal{A}_{\mathsf{Com}}$ breaks the binding property of the commitment scheme with probability $1/q^3(n)$, which is a contradiction. In particular,

$$\Pr\left[\mathsf{DecomPr}_{\neg\mathsf{Heaviest}}\right] < \frac{1}{8np(n)^2}. \tag{8}$$

Next, we claim that

$$\Pr\left[\mathsf{DecomPr}_{\mathsf{Heaviest}} \wedge \neg\mathsf{DecomPr}_{\neg\mathsf{Heaviest}})\right] < \frac{1}{7p(n)}.$$

In order to see this, observe that in the event $\{\mathsf{DecomPr}_{\mathsf{Heaviest}} \wedge \neg\mathsf{DecomPr}_{\neg\mathsf{Heaviest}}\}$ we have that

$$\mathsf{DecomPr}_{\mathsf{Heaviest}}(s[\rho^1_{B'}, \rho_{\mathcal{A}'}]) \leq \frac{1}{8p(n)} \quad \text{and} \quad \mathsf{DecomPr}_{\neg\mathsf{Heaviest}}(s[\rho^1_{B'}, \rho_{\mathcal{A}'}]) \leq \frac{1}{8np(n)^2}$$

This implies that

$$\mathsf{DecomPr}_{\{0,1\}^t}(s[\rho^1_{B'}, \rho_{\mathcal{A}'}]) \leq \frac{1}{8p(n)} + \frac{1}{8np(n)^2} < \frac{1}{7p(n)}$$

However, $\mathsf{DecomPr}_{\{0,1\}^t}(s[\rho^1_{B'}, \rho_{\mathcal{A}'}])$ just equals the probability that $\mathsf{decom}(s) \neq \perp$. Therefore, the above equation yields that $\Pr[\mathsf{IsGoodDef}(\text{Real})] < \frac{1}{7p(n)}$, and thus

$$\begin{aligned} &\Pr\left[\mathsf{DecomPr}_{\mathsf{Heaviest}} \wedge \neg\mathsf{DecomPr}_{\neg\mathsf{Heaviest}}\right] \\ &\quad\leq \Pr\left[\mathsf{IsGoodDef}(\text{Real}) \wedge \mathsf{DecomPr}_{\{0,1\}^t}(s[\rho^1_{B'}, \rho_{\mathcal{A}'}]) < \frac{1}{7p(n)}\right] < \frac{1}{7p(n)} \end{aligned} \tag{9}$$

Finally, it holds that

$$\begin{aligned} &\Pr\left[\mathsf{decom} \wedge \neg\mathsf{DecomPr}_{\neg\mathsf{Heaviest}}\right] \\ &\leq \Pr\left[\mathsf{decom}(\text{Real}) \in \neg\mathsf{Heaviest}(\text{Real}[\rho^1_{B'}, \rho_{\mathcal{A}'}]) \mid \mathsf{DecomPr}_{\neg\mathsf{Heaviest}}(\text{Real}[\rho^1_{B'}, \rho_{\mathcal{A}'}]) \leq \frac{1}{8np(n)^2}\right] \\ &\leq \frac{1}{8np(n)^2} \end{aligned} \tag{10}$$

Putting all of this together, we conclude that

$$\Pr\left[\mathsf{IsGoodDef}(\text{Real}) \wedge \text{Real} \notin L\right] \leq \frac{1}{8np(n)^2} + \frac{1}{7p(n)} + \frac{1}{8np(n)^2} < \frac{1}{6p(n)}$$

In order to complete the proof of this part, we need to prove that the above equation holds also with respect to Sim. We first note that for every $s \in \mathsf{Supp}(\text{Real})$ it holds that

$$\Pr[\text{Sim} = s] \leq \Pr[\text{Real} = s]. \tag{11}$$

In order to see that, observe that a sample $s = (x_A, x_B, \rho_{\mathcal{A}'}, \rho_{B'}, \mathsf{trans})$ drawn according to Real is fully determined by the value of $s(x_B, \rho_{\mathcal{A}'}, \rho_{B'})$, where this value is *uniformly* distributed over its domain. Thus, for every $s \in \mathsf{Supp}(\text{Real})$ it holds that $\Pr[\text{Real} = s] = \frac{1}{|\mathsf{Dom}_{x_B,\rho_{A'},\rho_{B'}}|}$, where $\mathsf{Dom}_{x_B,\rho_{A'},\rho_{B'}}$ contains all the possible values for $(x_B, \rho_{A'}, \rho_{B'})$ as induced by Real. On the other hand, it is easy to verify that $\Pr[\text{Sim}[x_B, \rho_{\mathcal{A}'}, \rho_{B'}] = (x_B, \rho_{\mathcal{A}'}, \rho_{B'})] \leq 1/|\mathsf{Dom}_{x_B,\rho_{A'},\rho_{B'}}|$ for every $(x_B, \rho_{\mathcal{A}'}, \rho_{B'}) \in \mathsf{Dom}_{x_B,\rho_{A'},\rho_{B'}}$, where the reason for the inequality is that the sampling stage of D may result in setting $z = \perp$, and the proof of Eq. (11) is concluded.

We conclude the proof of 1. by showing that

$$\Pr[\mathsf{IsGoodDef}(\text{Sim}) = 1 \wedge \text{Sim} \notin \mathsf{Supp}(\text{Real})] \leq \mathsf{neg}(n), \tag{12}$$

which together with Eq. (11) yield that

$$\begin{aligned}
&\Pr\left[\mathsf{IsGoodDef}(\text{Sim}) \wedge \text{Sim} \notin L\right] \\
&= \Pr\left[\mathsf{IsGoodDef}(\text{Sim}) \wedge \text{Sim} \notin L \wedge \text{Sim} \in \mathsf{Supp}(\text{Real})\right] \\
&\quad + \Pr\left[\mathsf{IsGoodDef}(\text{Sim}) \wedge \text{Sim} \notin L \wedge \text{Sim} \notin \mathsf{Supp}(\text{Real})\right] \\
&\leq \Pr\left[\mathsf{IsGoodDef}(\text{Real}) \wedge \text{Real} \notin L\right] + \Pr\left[\mathsf{IsGoodDef}(\text{Sim}) \wedge \text{Sim} \notin \mathsf{Supp}(\text{Real})\right] \\
&\leq \frac{1}{6p(n)} + \mathsf{neg}(n) < \frac{1}{4p(n)}\ .
\end{aligned}$$

For proving Eq. (12), we introduce a random variable $\mathsf{Decom}_{\mathsf{Sampl}}$ which is jointly distributed with Sim and is equal the decommitment of Com appears in the first valid defense given in the sampling stage of D induced by a random execution of (D, A, B) (it equals to $\perp$ if no valid defense is given). The definition of D yields that if $\mathsf{decom}(\text{Sim}) = \mathsf{Decom}_{\mathsf{Sampl}} \neq \perp$, then $\text{Sim} \in \mathsf{Supp}(\text{Real})$. Thus, it suffices to show that

$$\Pr[\mathsf{IsGoodDef}(\text{Sim}) = 1 \wedge \mathsf{decom}(\text{Sim}) \neq \mathsf{Decom}_{\mathsf{Sampl}}] \leq \mathsf{neg}(n)$$

As in the case of Eq. (8), it is easy to show that if the above does not holds, then Com is not binding.

Proving 2. Note that given Eq. (11), it suffices to prove that $(1 - \frac{1}{4p(n)}) \cdot \Pr[\text{Real} = s] \leq \Pr[\text{Sim} = s]$, for every $s \in L$.

We introduce another random variable $\mathsf{Init}_{\mathsf{Sampl}}$ jointly distributed with Sim, which is equal to the value of $(\rho_{\mathcal{A}'}, \rho^1_{B'})$ as chosen in the sampling stage of D induced by a random execution of (D, A, B). The following claim concludes the proof of 2., since both $\mathsf{Init}_{\mathsf{Sampl}}$ and $\text{Real}[\rho_{\mathcal{A}'}, \rho^1_{B'}]$ are uniformly distributed over the same set of values.

Claim 3.8 *For every $s \in L$ it holds that*

$$\Pr\left[\mathrm{Sim} = s \mid \mathsf{Init}_{\mathsf{Sampl}} = \mathsf{s}[\rho_{\mathcal{A}'}, \rho^1_{\mathsf{B}'}]\right] \geq (1 - \frac{1}{4p(n)}) \cdot \Pr\left[\mathrm{Real} = s \mid \mathrm{Real}[\rho_{\mathcal{A}'}, \rho^1_{B'}] = s[\rho_{\mathcal{A}'}, \rho^1_{B'}]\right]$$

Proof: We first note that since $\mathsf{DecomPr}_{\mathsf{Heaviest}}(s[\rho^1_{B'}, \rho_{\mathcal{A}'}]) \geq \frac{1}{8p(n)}$ (recall that $s \in L$), it holds that

$$\Pr\left[\mathsf{Decom}_{\mathsf{Sampl}} = \perp \mid \mathsf{Init}_{\mathsf{Sampl}} = \mathsf{s}[\rho_{\mathcal{A}'}, \rho^1_{\mathsf{B}'}]\right] < (1 - \frac{1}{8p(n)})^{n \cdot p(n)} = \mathsf{neg}(n).$$

Since $\mathsf{DecomPr}_{\neg\mathsf{Heaviest}}(s[\rho^1_{B'}, \rho_{\mathcal{A}'}]) \leq \frac{1}{8np(n)^2}$, it also holds that

$$\Pr\left[\mathsf{Decom}_{\mathsf{Sampl}} \in \neg\mathsf{Heaviest}(\rho^1_{B'}, \rho_{\mathcal{A}'}) \mid \mathsf{Init}_{\mathsf{Sampl}} = \mathsf{s}[\rho_{\mathcal{A}'}, \rho^1_{\mathsf{B}'}]\right] < n \cdot p(n) \cdot \frac{1}{8np(n)^2} = \frac{1}{8p(n)}$$

Hence,

$$\begin{aligned} &\Pr\left[\mathsf{Decom}_{\mathsf{Sampl}} = \mathsf{Heaviest}(\rho^1_{B'}, \rho_{\mathcal{A}'}) = \mathsf{decom}(s) \mid \mathsf{Init}_{\mathsf{Sampl}} = \mathsf{s}[\rho_{\mathcal{A}'}, \rho^1_{\mathsf{B}'}]\right] \qquad (13) \\ &> \quad 1 - \mathsf{neg}(n) - \frac{1}{8p(n)} > 1 - \frac{1}{4p(n)} \end{aligned}$$

Assuming that $\mathsf{Init}_{\mathsf{Sampl}} = \mathsf{s}[\rho_{\mathcal{A}'}, \rho^1_{\mathsf{B}'}]$, $\mathsf{Decom}_{\mathsf{Sampl}} = \mathsf{decom}(s)$ and $\mathrm{Sim}[x_B, \rho^2_{B'}, \rho^3_{B'}] = s[x_B, \rho^2_{B'}, \rho^3_{B'}]$, it is easy to verify that $\mathcal{A}'$ acts in Sim as A does in the underlying execution of π that generates Sim, and thus $\mathrm{Sim} = s$. It follows that

$$\begin{aligned} &\Pr\left[\mathrm{Sim} = s \mid \mathsf{Init}_{\mathsf{Sampl}} = \mathsf{s}[\rho_{\mathcal{A}'}, \rho^1_{\mathsf{B}'}] \wedge \mathsf{Decom}_{\mathsf{Sampl}} = \mathsf{decom}(\mathsf{s})\right] \\ &\quad \geq \Pr\left[\mathrm{Sim}[x_B, \rho^2_{B'}, \rho^3_{B'}] = s[x_B, \rho^2_{B'}, \rho^3_{B'}] \mid \mathsf{Init}_{\mathsf{Sampl}} = \mathsf{s}[\rho_{\mathcal{A}'}, \rho^1_{\mathsf{B}'}] \wedge \mathsf{Decom}_{\mathsf{Sampl}} = \mathsf{decom}(\mathsf{s})\right] \\ &\quad = \Pr\left[\mathrm{Real}[x_B, \rho^2_{B'}, \rho^3_{B'}] = s[x_B, \rho^2_{B'}, \rho^3_{B'}]\right] \\ &\quad = \Pr\left[\mathrm{Real} = s \mid \mathrm{Real}[\rho_{\mathcal{A}'}, \rho^1_{B'}] = s[\rho_{\mathcal{A}'}, \rho^1_{B'}]\right], \end{aligned}$$

where the second equality follows from the fact that under the given conditioning, $\mathrm{Sim}(x_B, \rho^2_{B'}, \rho^3_{B'})$ is uniformly distributed over the same set of possible values that $\mathrm{Real}[x_B, \rho^2_{B'}, \rho^3_{B'}]$ is uniformly distributed on. We conclude that

$$\begin{aligned} &\Pr\left[\mathrm{Sim} = s \mid \mathsf{Init}_{\mathsf{Sampl}} = \mathsf{s}[\rho_{\mathcal{A}'}, \rho^1_{\mathsf{B}'}] \wedge \mathsf{Decom}_{\mathsf{Sampl}} = \mathsf{decom}(\mathsf{s})\right] \\ &\geq \Pr\left[\mathsf{Decom}_{\mathsf{Sampl}} = \mathsf{decom}(s) \mid \mathsf{Init}_{\mathsf{Sampl}} = \mathsf{s}[\rho_{\mathcal{A}'}, \rho^1_{\mathsf{B}'}]\right] \cdot \Pr\left[\mathrm{Real} = s \mid \mathrm{Real}[\rho_{\mathcal{A}'}, \rho^1_{B'}] = s[\rho_{\mathcal{A}'}, \rho^1_{B'}]\right] \\ &\geq (1 - \frac{1}{4p(n)}) \cdot \Pr\left[\mathrm{Real} = s \mid \mathrm{Real}[\rho_{\mathcal{A}'}, \rho^1_{B'}] = s[\rho_{\mathcal{A}'}, \rho^1_{B'}]\right]. \end{aligned}$$

■

This completes the proof of Claim 3.7. ■

Applying Lemma 3.2 twice, once with $A = P_1$ and $B = P_2$ (and $g_A = g_1$,$g_B = g_2$) and another time with $A = P_2$ and $B = P_1$ (and $g_A = g_2$,$g_B = g_1$), we obtain that π' is defensibly private with respect to (g_1, g_2) if π was semi-honest private with respect to (g_1, g_2). This completes the proof of Theorem 3.1. ■

We have the following corollary of Theorem 3.1:

Corollary 3.9 *There exists a fully black-box reduction from defensibly private oblivious transfer to semi-honest private oblivious transfer.*

Proof: Theorem 3.1 yields the existence of a fully black-box reduction from defensibly private oblivious transfer to semi-honest private oblivious transfer and one-way functions. Thus, the proof of the corollary follows from the fact that there exists a fully black-box reduction from one-way functions to semi-honest oblivious transfer. A proof of this folklore fact can be found in Appendix B. ■

4 Achieving Security Against a Malicious Receiver

In this section we transform any defensibly private oblivious transfer protocol into a bit oblivious transfer protocol that is secure in the presence of a malicious receiver and private in the presence of a defensible sender. We stress that the security achieved for malicious receivers is according to the ideal/real model definition of security for secure computation. Our construction uses black-box access to an oblivious transfer protocol that is private for defensible receivers and senders (like the one given by Corollary 3.9). That is, we prove the following theorem:

Theorem 4.1 *There exists a fully black-box reduction from oblivious transfer that is secure in the presence of a malicious receiver and private in the presence of a defensible sender to oblivious transfer that is defensibly private.*

Stated differently, in this section we show how to boost the security guarantee from *privacy in the presence of a defensible receiver* to *security in the presence of a malicious receiver*. The guarantee regarding a corrupted sender remains unchanged. In order to prove this theorem (and the next), we find it more convenient to use the alternative and more explicit definition of defensible oblivious transfer found in Section 2.5.1.

We begin by presenting the protocol:

Protocol 4.2 (bit oblivious transfer secure in the presence of malicious receivers):

- **Inputs:** *The sender S has a pair of bits (s_0, s_1) for input and the receiver R has a bit r.*
- **The protocol:**
  1. *The receiver R chooses $2n$ random bits $r_1, \ldots, r_{2n} \in_R \{0,1\}$.*
  2. *The sender S chooses $2n$ pairs of random bits $s_i^0, s_i^1 \in_R \{0,1\}$ for $i = 1, \ldots, 2n$.*
  3. *S and R run $2n$ parallel executions of a bit oblivious transfer protocol π that is private in the presence of defensible receivers and defensible senders. In the i^{th} execution, S inputs (s_i^0, s_i^1) and R inputs r_i. Let $t_1, \ldots, t_{2n}$ be the transcripts that result from these executions.*
  4. *S and R run a secure two-party coin-tossing protocol (that accesses a one-way function in a black-box way) for generating a random string $q = q_1, \ldots, q_n$ of length n.[7] The string q is used to define a set of indices $Q \subset \{1, \ldots, 2n\}$ of size n in the following way: $Q = \{2i - q_i\}_{i=1}^n$. (Thus, for $n = 3$ and $q = 010$ we have that $Q = \{2, 3, 6\}$.)*

[7] Sequential executions of the coin-tossing protocol of [3] can be used. The fact that this is a fully black-box reduction from secure coin-tossing to bit commitment (hence, via [19, 32] to one-way functions) is proved in [13].

5. *For every $i \in Q$, the receiver R provides a defense (r_i, ρ_r^i).*
6. *S checks that for every $i \in Q$, the pair (r_i, ρ_r^i) constitutes a good defense by R for t_i. If not, then S aborts and halts. Otherwise, it continues to the next step.*
7. *For every $j \notin Q$, the receiver R computes $\alpha_j = r \oplus r_j$ (where r is R's initial input) and sends $\{\alpha_j\}_{j \notin Q}$ to S.*
8. *S computes $\sigma_0 = s_0 \oplus \left(\bigoplus_{j \notin Q} s_j^{\alpha_j}\right)$ and $\sigma_1 = s_1 \oplus \left(\bigoplus_{j \notin Q} s_j^{1-\alpha_j}\right)$, and sends (σ_0, σ_1) to R.*
9. *R computes and outputs $s_r = \sigma_r \oplus \left(\bigoplus_{j \notin Q} s_j^{r_j}\right)$.*

We stress that our proof below of Protocol 4.2 relies on the fact that the sender's inputs are single bits.[8]

Claim 4.3 *Assume that π is a non-trivial oblivious transfer protocol that is private for random inputs in the presence of defensible senders and receivers. Then, Protocol 4.2 is a non-trivial oblivious transfer protocol that is* secure in the presence of malicious receivers *and private in the presence of defensible senders.*

Proof: We first demonstrate the non-triviality property; that is, we show that if S and R are honest, then R receives s_r, as required. To see this, first note that by the non-triviality of π, the receiver R obtains all of the bits $s_j^{r_j}$, and in particular all $s_j^{r_j}$ for $j \notin Q$. Now, if $r = 0$, then R sets $\alpha_j = r_j$ for every j. Therefore, R will compute $s_0 = \sigma_0 \oplus \left(\bigoplus_{j \notin Q} s_j^{r_j}\right) = \sigma_0 \oplus \left(\bigoplus_{j \notin Q} s_j^{\alpha_j}\right)$. This computation is correct because S computed $\sigma_0 = s_0 \oplus \left(\bigoplus_{j \notin Q} s_j^{\alpha_j}\right)$. In contrast, if $r = 1$, then $\alpha_j = 1 \oplus r_j$ for every j, which is equivalent to $r_j = 1 - \alpha_j$. Thus, once again, R's computation of $\bigoplus_{j \notin Q} s_j^{r_j}$ when computing s_1 equals S's computation of $\bigoplus_{j \notin Q} s_j^{1-\alpha_j}$ when computing σ_1, and R will obtain σ_1.

Privacy in the presence of defensible senders. We now prove that Protocol 4.2 is private in the presence of a defensible sender $\mathcal{A}$. Intuitively, if protocol π is private in the presence of a defensible sender, then a defensible adversary here cannot learn any of the r_i values in the execution (apart from those explicitly revealed by R when it provides its defenses). Therefore, the $\alpha_j = r_j \oplus r$ values that it receives reveal nothing of the receiver's input r, because for all $j \notin Q$, the value r_j is not learned. Before we prove this formally, we define the following experiment that considers n executions of the subprotocol π:

Experiment $\mathsf{Expt}_\pi^n(\mathcal{A}_n, b)$:

1. *Choose $r = r_1, \ldots, r_n \in_R \{0,1\}^n$ uniformly at random, and give r to $\mathcal{A}_n$.*
2. *Run n executions of π with $\mathcal{A}_n$ as the sender and the honest R as the receiver. In the i^{th} execution, R uses input $r_i \oplus b$. (Thus, the inputs of R in all executions are either r or its complement.)*

[8] This is due to our definition of "oblivious transfer that is private for defensible adversaries". It is possible to define a stronger notion of defensible adversaries that *is* sufficient for proving that Protocol 4.2 is secure even when the sender's inputs are strings of an arbitrary length. Nevertheless, the case of bit oblivious transfer is simpler and suffices for proving our result.

3. *S outputs defenses for all executions and a guess τ.*

4. *If all the defenses are good, output τ. Otherwise, output $\perp$.*

We now show that if π is private in the presence of a defensible sender (as defined with respect to $\mathsf{Expt}^{\mathrm{snd}}_{\pi}$), then for every adversary $\mathcal{A} = \{\mathcal{A}_n\}$, every polynomial $p(\cdot)$, and all sufficiently large n's

$$\left|\Pr\left[\mathsf{Expt}^{n}_{\pi}(\mathcal{A}_n, 1) = 1\right] - \Pr\left[\mathsf{Expt}^{n}_{\pi}(\mathcal{A}_n, 0) = 1\right]\right| < \frac{1}{p(n)} \tag{14}$$

Assume by contradiction that there exists an adversary $\mathcal{A}$ and polynomial $p(\cdot)$ for which the above does not hold. We use these to construct a defensible adversary $\hat{\mathcal{A}}$ that succeeds in $\mathsf{Expt}^{\mathrm{snd}}_{\pi}$ with polynomial advantage. We prove this using a hybrid argument. For ℓ such that $0 \leq \ell \leq n$, define by $H_\ell(\mathcal{A}_n)$ an experiment with $\mathcal{A}_n$ defined exactly as above, except that the receiver R uses input r_i for $i \leq \ell$, and input $r_i \oplus 1$ for $i > \ell$; the output is defined in the same way. We have that $H_0(\mathcal{A}_n) = \mathsf{Expt}^{n}_{\pi}(\mathcal{A}_n, 1)$ and $H_n(\mathcal{A}_n) = \mathsf{Expt}^{n}_{\pi}(\mathcal{A}_n, 0)$. Thus, by our contradicting assumption $|\Pr[H_0(\mathcal{A}_n) = 1] - \Pr[H_n(\mathcal{A}_n) = 1]| \geq 1/p(n)$. By the triangle inequality, there exists a value ℓ $(0 \leq \ell < n)$ such that

$$\left|\Pr[H_\ell(\mathcal{A}_n) = 1] - \Pr[H_{\ell+1}(\mathcal{A}_n) = 1]\right| \geq \frac{1}{np(n)}$$

We use this fact in order to construct a defensible adversary $\hat{\mathcal{A}} = \{\hat{\mathcal{A}}_n\}$ who attacks the protocol π. Adversary $\hat{\mathcal{A}}_n$ internally invokes $\mathcal{A}_n$ and simulates an execution of the experiment in the following way. $\hat{\mathcal{A}}_n$ chooses a random $r = r_1, \ldots, r_n$ and runs the honest R in all executions except for the $(\ell+1)^{\text{th}}$ one; the messages of the $(\ell+1)^{\text{th}}$ execution are forwarded externally from $\mathcal{A}_n$ to the real honest receiver that $\hat{\mathcal{A}}_n$ interacts with (for a single execution of π). For the internally simulated executions, $\hat{\mathcal{A}}_n$ uses inputs $r_1, \ldots, r_\ell$ for executions 1 to ℓ, respectively, and inputs $r_{\ell+2} \oplus 1, \ldots, r_n \oplus 1$ for executions $\ell+2$ to n, respectively. At the end of all executions $\hat{\mathcal{A}}_n$ obtains $\mathcal{A}$'s outputs (that include its defenses and a guess τ). If any of the defenses are not good, then $\hat{\mathcal{A}}_n$ outputs $\perp$. Otherwise, if all the defenses are good, $\hat{\mathcal{A}}_n$ outputs τ.

Now, notice that if the real external R's input equals $r_{\ell+1}$ then the view of $\mathcal{A}_n$ is identical to $H_\ell(\mathcal{A}_n)$. In contrast, if the real external R's input equals $r_{\ell+1} \oplus 1$ then the view of $\mathcal{A}_n$ is identical to $H_{\ell+1}(\mathcal{A}_n)$. It therefore follows that

$$\Pr\left[\mathsf{Expt}^{\mathrm{snd}}_{\pi}(\hat{\mathcal{A}}_n, r_{\ell+1}) = 1\right] = \Pr\left[H_\ell(\mathcal{A}_n) = 1\right]$$

and

$$\Pr\left[\mathsf{Expt}^{\mathrm{snd}}_{\pi}(\hat{\mathcal{A}}_n, r_{\ell+1} \oplus 1) = 1\right] = \Pr\left[H_{\ell+1}(\mathcal{A}_n) = 1\right]$$

Since $r_{\ell+1}$ equals 0 if and only if $r_{\ell+1} \oplus 1 = 1$, we have that

$$\begin{aligned}\left|\Pr\left[\mathsf{Expt}^{\mathrm{snd}}_{\pi}(\hat{\mathcal{A}}_n, 1) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathrm{snd}}_{\pi}(\hat{\mathcal{A}}_n, 0) = 1\right]\right| \\ = \left|\Pr[H_\ell(\mathcal{A}_n) = 1] - \Pr[H_{\ell+1}(\mathcal{A}_n) = 1]\right| \geq \frac{1}{np(n)}\end{aligned}$$

in contradiction to the assumption that π is private in the presence of defensible senders.

We now use Eq. (14) in order to prove that Protocol 4.2 is private in the presence of defensible senders. We will present a proof in the "hybrid model", where the parties have access to a trusted

party computing the coin-tossing protocol. The soundness of this methodology follows from the sequential composition theorems in [4, 13] and the fact that the coin-tossing protocol we employ in Step 4 of Protocol 4.2 is secure under the simulation-based security definitions from [4, 13].

Assume, by contradiction, that there exists an adversary $\mathcal{A}$ and a polynomial $p(\cdot)$ such that for infinitely many n's

$$\left|\Pr\left[\mathsf{Expt}_{\Pi}^{\mathrm{snd}}(\mathcal{A}_n,1)=1\right]-\Pr\left[\mathsf{Expt}_{\Pi}^{\mathrm{snd}}(\mathcal{A}_n,0)=1\right]\right|\geq\frac{1}{p(n)}$$

where Π denotes Protocol 4.2 (not to be confused with π that denotes the subprotocol). Then, we construct an adversary $\hat{\mathcal{A}}=\{\hat{\mathcal{A}}_n\}$ that contradicts Eq. (14).

The adversary $\hat{\mathcal{A}}_n$ interacts with an external R in Expt_{π}^{n}; let r be the n-bit string that $\hat{\mathcal{A}}_n$ receives in the context of this experiment. Now, $\hat{\mathcal{A}}_n$ internally invokes $\mathcal{A}_n$ and simulates an execution of Protocol 4.2 for $\mathcal{A}_n$, as follows. First, $\hat{\mathcal{A}}_n$ chooses a random n-bit string q and defines the index set Q as specified in the protocol. Then, $\hat{\mathcal{A}}_n$ runs $2n$ executions of π with $\mathcal{A}_n$: for every $i\in Q$, adversary $\hat{\mathcal{A}}_n$ internally plays the honest receiver R in the i^{th} execution of π with a random input r_i; for every $j\notin Q$, adversary $\hat{\mathcal{A}}_n$ forwards the messages of the j^{th} execution between the external receiver and $\mathcal{A}_n$. After the executions of π have concluded, $\hat{\mathcal{A}}_n$ hands $\mathcal{A}_n$ the string q as if coming from the trusted party who computes the coin-tossing functionality for the parties (in the hybrid model). Given that q and thus Q is determined, $\hat{\mathcal{A}}_n$ hands $\mathcal{A}_n$ good defenses for every execution $i\in Q$ ($\hat{\mathcal{A}}_n$ can do this because these are exactly the executions that it ran internally with $\mathcal{A}_n$). Finally $\hat{\mathcal{A}}_n$ defines the α_j values to hand to $\mathcal{A}_n$. These values are computed from the string $r=r_1,\ldots,r_n$ that $\hat{\mathcal{A}}_n$ received in its experiment Expt_{π}^{n}. Let $\alpha_{j_1},\ldots,\alpha_{j_n}$ be the n values of $j_i\notin Q$. Then, $\hat{\mathcal{A}}_n$ defines $\alpha_{j_i}=r_i$. Finally, $\hat{\mathcal{A}}_n$ receives $\mathcal{A}_n$'s defense and guess τ, and outputs this same pair.

First, notice that if the input of the honest R that $\hat{\mathcal{A}}_n$ interacts with in Expt_{π}^{n} is $r=r_1,\ldots,r_n$, then $\mathcal{A}_n$'s view is identical to its view in Protocol 4.2 where the receiver has input 0 (because each $\alpha_{j_i}=r_j$ where r_j is the input used in the j_i^{th} execution of π and $j_i\notin Q$). Next, if the input of the honest R that $\hat{\mathcal{A}}_n$ interacts with in Expt_{π}^{n} is $r_1\oplus 1,\ldots,r_n\oplus 1$, then $\mathcal{A}_n$'s view is identical to its view in Protocol 4.2 where the receiver has input 1 (because each $\alpha_{j_i}=1\oplus r_j$ where r_j is the input used in the j_i^{th} execution of π and $j_i\notin Q$). Thus, for $b\in\{0,1\}$,

$$\Pr\left[\mathsf{Expt}_{\pi}^{n}(\hat{\mathcal{A}}_n,b)=1\right]=\Pr\left[\mathsf{Expt}_{\Pi}^{\mathrm{snd}}(\mathcal{A}_n,b)=1\right]$$

and so by the contradicting assumption, for infinitely many n's

$$\left|\Pr\left[\mathsf{Expt}_{\pi}^{n}(\hat{\mathcal{A}}_n,1)=1\right]-\Pr\left[\mathsf{Expt}_{\pi}^{n}(\hat{\mathcal{A}}_n,0)=1\right]\right|\geq\frac{1}{p(n)}$$

which contradicts Eq. (14). We remark that this proof holds irrespective of the scheduling of the $2n$ executions of π. Therefore, for efficiency reasons, they can be run in parallel (as described in the protocol itself).

Security in the presence of malicious receivers. We now prove that Protocol 4.2 is secure in the presence of malicious receivers. The intuition behind this proof is that the cut-and-choose technique forces an adversarial receiver $\mathcal{A}$ to be able to provide a good defense for most of the oblivious transfer executions (or be caught with overwhelming probability). In particular, there must be at least one $j\notin Q$ for which $\mathcal{A}$ *could* have provided a good defense. This implies that there

exists an index j for which $\mathcal{A}$ cannot predict the value of $s_j^{1-r_j}$ with any non-negligible advantage. Since s_{1-r} is masked by $s_j^{1-r_j}$, it follows that $\mathcal{A}$ also learns nothing about s_{1-r}. We stress that the above intuition shows that a malicious $\mathcal{A}$ cannot learn anything about s_{1-r}. However, we actually need to prove a much stronger claim in that the protocol is *secure* for a malicious R^*, as defined via the ideal/real model simulation paradigm. As above, we present our analysis in the hybrid model, where the honest parties use a trusted party to compute the coin-tossing functionality for them.

We now describe the simulator Sim (given black-box access to the adversary $\mathcal{A} = \{\mathcal{A}_n\}$):

1. For each $i = 1, \ldots, 2n$, simulator Sim chooses random pairs $s_i^0, s_i^1 \in_R \{0,1\}$ and plays the honest sender in π with these inputs, where $\mathcal{A}_n$ plays the receiver.

2. Sim chooses a random string $q \in_R \{0,1\}^n$ and hands it to $\mathcal{A}_n$ as if it is the output of the coin-tossing functionality, as sent by the trusted party. Let Q be the index set derived from q. Upon receiving back pairs (r_i, ρ_r^i) for $i \in Q$, simulator Sim checks that they all constitute good defenses, respectively. If not, then it aborts (just like the honest sender).

3. Sim rewinds $\mathcal{A}_n$ to the beginning of the previous step and chooses a new random string q' with associated index set Q'. (We stress that the choice of q' is independent of the choice of q.) Sim hands q' to $\mathcal{A}_n$ and sees if it replies with pairs (r_i, ρ_r^i) that are good defenses, for all $i \in Q'$. Sim repeats this process with a new q' until $\mathcal{A}_n$ indeed replies with pairs (r_i, ρ_r^i) that are good defenses, for all $i \in Q'$. If $Q' = Q$, then Sim outputs fail. Otherwise it proceeds to the next step.

4. Given that $Q' \neq Q$ (and $|Q'| = |Q|$), there exists at least one index j such that $j \notin Q'$ but $j \in Q$. For such a j, Sim computes $r = r_j \oplus \alpha_j$ and sends r to the trusted party. (Note that r_j is obtained from the defense (r_j, ρ_r^j) that was received from $\mathcal{A}_n$ after it was sent the query set Q. In contrast, α_j is the value received from $\mathcal{A}_n$ after rewinding; i.e., when the query set was Q'.)

5. Upon receiving back a bit s_r from the trusted party, Sim computes σ_0 and σ_1 as follows:

   (a) If $r = 0$, then $\sigma_0 = s_0 \oplus \left(\bigoplus_{j \notin Q'} s_j^{\alpha_j} \right)$ and $\sigma_1 \in_R \{0,1\}$.

   (b) If $r = 1$, then $\sigma_0 \in_R \{0,1\}$ and $\sigma_1 = s_1 \oplus \left(\bigoplus_{j \notin Q'} s_j^{1-\alpha_j} \right)$.

   Sim sends (σ_0, σ_1) to $\mathcal{A}_n$ and outputs whatever $\mathcal{A}_n$ does.

We now prove that the joint output of Sim and the honest sender S in the ideal model is computationally indistinguishable from the joint output of $\mathcal{A}_n$ and S in the real model. Actually, since the honest S has no output from the protocol, it suffices here to show that the output of Sim in the ideal model is computationally indistinguishable from the output of $\mathcal{A}_n$ in the real model.

We first claim that apart from the pair (σ_0, σ_1), the view of $\mathcal{A}_n$ in the simulation with Sim is *statistically close* to its view in a real execution with S; the only difference being in the case that Sim outputs fail. This can be seen as follows: if $\mathcal{A}_n$ does not send good defenses after receiving q, then Sim aborts, just as the honest S would (and in this case the simulation is perfect). If $\mathcal{A}_n$ does send good defenses, then Sim continues until it finds another (independent) q' for which $\mathcal{A}_n$ also replies with good defenses. This strategy can be described as follows: uniformly sample a partial transcript from the distribution $\langle S, \mathcal{A}_n \rangle$ to see if the event X occurs (the event X here

is $\mathcal{A}_n$ replying correctly and the partial transcript is until the point that (σ_0, σ_1) is sent); if yes, then sample a partial transcript from amongst those for which X occurs. It is immediate that the resulting partial transcript is uniformly distributed among all partial transcripts. The only deviation from the above strategy occurs if $q' = q$, in which case Sim outputs fail. In order to see that this occurs with only negligible probability, let ϵ denote the probability that $\mathcal{A}_n$ replies correctly upon receiving a random string q of length n, as the output from the coin-tossing protocol. Since there are 2^n such strings, it follows that $\mathcal{A}_n$ replies correctly on $\epsilon \cdot 2^n$ of these strings. Now, the probability that Sim outputs fail equals the probability that $\mathcal{A}_n$ replied correctly to the first string q, times the probability that $q' = q$, where this latter probability is $1/\epsilon 2^n$ because there are $\epsilon 2^n$ strings upon which $\mathcal{A}_n$ replies correctly. Thus:

$$\Pr[\mathsf{Sim} \text{ outputs } \mathsf{fail}] = \epsilon \cdot \frac{1}{\epsilon \cdot 2^n} = \frac{1}{2^n}$$

which is negligible.

Until now, we have shown that in the simulation by Sim, the adversary $\mathcal{A}_n$'s partial view up until the point that it receives (σ_0, σ_1) is statistically close to its view in a real execution with S. We now show that $\mathcal{A}_n$'s full view is computationally indistinguishable. To do this, we consider a modified ideal-model simulator Sim' who receives the sender S's input pair (s_0, s_1). Simulator Sim' works in exactly the same way as Sim, except that it computes σ_{1-r} as an honest sender would instead of choosing it uniformly. By the above argument, it follows that the distribution generated by Sim' in the ideal model is *statistically close* to the distribution generated by a real execution between S and $\mathcal{A}_n$. (Recall that Sim already generates σ_r in the same way as an honest S, and therefore so does Sim'.) It remains to show that the distribution generated by Sim' is computationally indistinguishable from that generated by Sim.

The only difference between Sim' and Sim is in the generation of σ_{1-r}: simulator Sim' generates it "honestly", whereas Sim chooses it uniformly. As mentioned above, intuitively, indistinguishability follows from the fact that at least one $s_j^{1-r_j}$ masks the value of s_{1-r}. Formally, we show that if this "fake" σ_{1-r} can be distinguished from a real one, then we can construct a defensible receiver $\tilde{\mathcal{A}}_n$ that can break the oblivious transfer protocol π.

That is, we show that if the output generated by Sim and Sim' can be distinguished with non-negligible probability, then it is possible for a *defensible adversary* $\tilde{\mathcal{A}}_n$ to succeed in the experiment of Definition 2.9 with non-negligible advantage, with respect to the subprotocol π. Assume by contradiction that there exists a distinguisher D, a polynomial $p(\cdot)$ and infinitely many n's such that

$$|\Pr[D(\mathsf{output}_{\mathsf{Sim}}) = 1] - \Pr[D(\mathsf{output}_{\mathsf{Sim}'}) = 1]| \geq \frac{1}{p(n)} \ .$$

Without loss of generality, assume that

$$\Pr[D(\mathsf{output}_{\mathsf{Sim}}) = 1] - \Pr[D(\mathsf{output}_{\mathsf{Sim}'}) = 1] \geq \frac{1}{p(n)} \ . \tag{15}$$

We now use the above to construct a defensible adversary $\tilde{\mathcal{A}} = \{\tilde{\mathcal{A}}_n\}$. Adversary $\tilde{\mathcal{A}}_n$ begins its attack by starting the simulation of Protocol 4.2, according to Sim's strategy. Specifically, $\tilde{\mathcal{A}}_n$ chooses $s_0, s_1 \in_R \{0,1\}$ and runs the simulation strategy of Sim with $\mathcal{A}_n$ up until the point where σ_0 and σ_1 are sent. The simulation is the same as Sim, except for the following difference: $\tilde{\mathcal{A}}_n$ begins by choosing $j \in_R \{1, \ldots, 2n\}$ and internally invokes $\mathcal{A}_n$, simulating an execution of

Protocol 4.2. Then, all of the oblivious transfer subexecutions of π, *except* for the j^{th} one, are run internally with $\tilde{\mathcal{A}}_n$ playing the honest sender ($\tilde{\mathcal{A}}_n$ also chooses the s_i^0 and s_i^1 values as S would); in contrast, the messages of the j^{th} execution of the oblivious transfer protocol π are forwarded between $\tilde{\mathcal{A}}_n$'s external sender and the internal $\mathcal{A}_n$ playing the receiver. Following the oblivious transfer executions, $\tilde{\mathcal{A}}_n$ runs the honest sender in the coin-tossing protocol to generate q and thus Q as required. If $j \notin Q$, then $\tilde{\mathcal{A}}_n$ outputs fail and halts. Otherwise, $\tilde{\mathcal{A}}_n$ receives back the defenses; since $j \in Q$, the j^{th} defense is included. If (r_j, ρ_r^j) is not a good defense, then $\tilde{\mathcal{A}}_n$ outputs fail and halts. Otherwise, it stores (r_j, ρ_r^j) and continues like Sim by rewinding $\mathcal{A}_n$ and generating a new q' and Q'. If $j \in Q'$, then once again $\tilde{\mathcal{A}}_n$ outputs fail and halts. Otherwise, it continues like Sim (using the j chosen above for which it is given that $j \in Q$ and $j \notin Q'$). $\tilde{\mathcal{A}}_n$ continues in the same way that Sim does up until (but not including) the point at which (σ_0, σ_1) must be sent. Now, $\tilde{\mathcal{A}}_n$ computes (σ_0, σ_1) as follows. First, note that $\tilde{\mathcal{A}}_n$ knows the values (s_0, s_1) and s_i^0, s_i^1 for all $i \neq j$ (because it chose them). However, the values s_j^0 and s_j^1 are not known to $\tilde{\mathcal{A}}_n$ because these are the values used by the external sender with whom it interacts. Nevertheless, the (good) defense provided by $\mathcal{A}_n$ is enough to obtain the value $s_j^{r_j}$. This holds because given the transcript of the j^{th} oblivious transfer execution and the input and random-tape of the receiver, it is possible to derive $s_j^{r_j}$. The only value unknown to $\tilde{\mathcal{A}}_n$ is therefore $s_j^{1-r_j}$. Therefore, $\tilde{\mathcal{A}}_n$ is able to compute σ_r like the honest sender. In contrast, it cannot honestly compute σ_{1-r}. Rather, $\tilde{\mathcal{A}}_n$ guesses the value of $s_j^{1-r_j} \in_R \{0,1\}$ randomly, and then computes σ_{1-r} using s_{1-r}, all of the s_i values that it knows (i.e., all apart from $s_j^{1-r_j}$), and the uniformly chosen $s_j^{1-r_j}$.

In order to determine its output, $\tilde{\mathcal{A}}_n$ obtains the output of $\mathcal{A}_n$ and runs the distinguisher D (from Eq. (15)) on this output; let b be the bit output by D. Then, $\tilde{\mathcal{A}}_n$ sets $\tau = s_j^{1-r_j} \oplus b$. (Recall that τ is $\tilde{\mathcal{A}}_n$'s guess for the "not-received" bit used by the honest sender. The motivation for this guess is that by Eq. (15), D outputs 1 with higher probability on Sim (when the bit is random) than on Sim' (when the bit is correct). Thus, when D outputs 1, we flip $\tilde{\mathcal{A}}_n$'s guess for $s_j^{1-r_j}$.) Finally, $\tilde{\mathcal{A}}_n$ outputs the defense (r_j, ρ_r^j) obtained above and the bit τ.

We proceed to analyze the probability that $\tilde{\mathcal{A}}_n$ succeeds in $\mathsf{Expt}_\pi^{\text{rec}}$. First, note that unless $\tilde{\mathcal{A}}_n$ outputs fail, the view of $\mathcal{A}_n$ when interacting with $\tilde{\mathcal{A}}_n$ above is *identical* to its view in the simulation by Sim. This is due to the fact that $\tilde{\mathcal{A}}_n$ follows Sim's strategy, except for two differences. The first difference is that in the j^{th} execution of the oblivious transfer protocol π is run externally. However, since Sim plays the role of an honest receiver in all of the executions, this makes no difference to $\mathcal{A}_n$'s view. The second difference is in how σ_{1-r} is computed: Sim chooses it uniformly, whereas $\tilde{\mathcal{A}}_n$ computes it as described above. Clearly, the distribution generated is the same because $\tilde{\mathcal{A}}_n$ uses a uniformly distributed $s_j^{1-r_j}$, and thus σ_{1-r} is also uniformly distributed.

Now, denote the inputs of the honest sender that $\tilde{\mathcal{A}}_n$ interacts with by $(\tilde{s}_0, \tilde{s}_1)$. Using the facts that **(a)** $\tilde{\mathcal{A}}_n$ generates the exact same distribution as Sim, **(b)** $\tilde{\mathcal{A}}_n$ sets $\tau = s_j^{1-r_j} \oplus b$ (where b is D's output bit), and **(c)** $\tilde{\mathcal{A}}_n$ presents a good defense every time that it does not output fail, we have that

$$\Pr\left[\mathsf{Expt}_\pi^{\text{rec}}(\tilde{\mathcal{A}}_n) = 1 \mid \mathsf{output}_{\tilde{\mathcal{A}}_n} \neq \mathsf{fail}\right] = \Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) \oplus s_j^{1-r_j} = \tilde{s}_{1-r_j}\right] .$$

(Recall that $\mathsf{Expt}_\pi^{\text{rec}}(\tilde{\mathcal{A}}_n) = 1$ if $\tilde{\mathcal{A}}_n$ presents a good defense and $\tau = \tilde{s}_{1-r_j}$.)

In contrast to the above, conditioned on the event that $s_j^{1-r_j} = \tilde{s}_{1-r_j}$ (i.e., the event that $\tilde{\mathcal{A}}_n$ guessed correctly), the result is an execution that is distributed exactly according to Sim'. (Recall

that the only difference between Sim and Sim' is with respect to the computation of σ_{1-r}.) That is,

$$\begin{aligned}
&\Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) \oplus s_j^{1-r_j} = \tilde{s}_{1-r_j} \mid s_j^{1-r_j} = \tilde{s}_{1-r_j}\right] \\
&\quad = \Pr\left[D(\mathsf{output}_{\mathsf{Sim}'}) \oplus s_j^{1-r_j} = \tilde{s}_{1-r_j} \mid s_j^{1-r_j} = \tilde{s}_{1-r_j}\right] \\
&\quad = \Pr\left[D(\mathsf{output}_{\mathsf{Sim}'}) = 0\right]
\end{aligned}$$

where the last equality is just due to the fact that $s_j^{1-r_j} = \tilde{s}_{1-r_j}$. Now, recalling that $s_j^{1-r_j}$ is chosen uniformly by $\tilde{\mathcal{A}}_n$ (and so equals $\tilde{s}_{1-r_j}$ with probability exactly $1/2$), we have:

$$\begin{aligned}
&\Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) \oplus s_j^{1-r_j} = \tilde{s}_{1-r_j}\right] \\
&= \frac{1}{2} \cdot \Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) \oplus s_j^{1-r_j} = \tilde{s}_{1-r_j} \mid s_j^{1-r_j} = \tilde{s}_{1-r_j}\right] \\
&\quad + \frac{1}{2} \cdot \Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) \oplus s_j^{1-r_j} = \tilde{s}_{1-r_j} \mid s_j^{1-r_j} \neq \tilde{s}_{1-r_j}\right] \\
&= \frac{1}{2} \cdot \Pr\left[D(\mathsf{output}_{\mathsf{Sim}'}) = 0\right] + \frac{1}{2} \cdot \Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) = 1 \mid s_j^{1-r_j} \neq \tilde{s}_{1-r_j}\right] \\
&= \frac{1}{2} \cdot \left(1 - \Pr\left[D(\mathsf{output}_{\mathsf{Sim}'}) = 1\right]\right) + \frac{1}{2} \cdot \Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) = 1 \mid s_j^{1-r_j} \neq \tilde{s}_{1-r_j}\right] \\
&= \frac{1}{2} + \frac{1}{2} \cdot \Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) = 1 \mid s_j^{1-r_j} \neq \tilde{s}_{1-r_j}\right] - \frac{1}{2} \cdot \Pr\left[D(\mathsf{output}_{\mathsf{Sim}'}) = 1\right] \ .
\end{aligned}$$

Recalling again that when $s_j^{1-r_j} = \tilde{s}_{1-r_j}$ the output of Sim is the same as Sim', we have that

$$\begin{aligned}
&\frac{1}{2} + \frac{1}{2} \cdot \Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) = 1 \mid s_j^{1-r_j} \neq \tilde{s}_{1-r_j}\right] - \frac{1}{2} \cdot \Pr\left[D(\mathsf{output}_{\mathsf{Sim}'}) = 1\right] \\
&\quad = \frac{1}{2} + \frac{1}{2} \cdot \Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) = 1 \mid s_j^{1-r_j} = \tilde{s}_{1-r_j}\right] + \frac{1}{2} \cdot \Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) = 1 \mid s_j^{1-r_j} \neq \tilde{s}_{1-r_j}\right] \\
&\qquad - \Pr\left[D(\mathsf{output}_{\mathsf{Sim}'}) = 1\right] \\
&\quad = \frac{1}{2} + \Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) = 1\right] - \Pr\left[D(\mathsf{output}_{\mathsf{Sim}'}) = 1\right] \ .
\end{aligned}$$

Combining the above with Eq. (15), we have that for infinitely many n's

$$\Pr\left[\mathsf{Expt}_{\pi}^{\mathrm{rec}}(\tilde{\mathcal{A}}_n) = 1 \mid \mathsf{output}_{\tilde{\mathcal{A}}_n} \neq \mathsf{fail}\right] = \Pr\left[D(\mathsf{output}_{\mathsf{Sim}}) \oplus s_j^{1-r_j} = \tilde{s}_{1-r_j}\right] \geq \frac{1}{2} + \frac{1}{p(n)} \ .$$

Recall now that $\tilde{\mathcal{A}}_n$ outputs fail if $\mathcal{A}_n$ does not output a good defense, if $j \notin Q$, or if $j \in Q'$. We first claim that $\mathcal{A}_n$ must output a good defense with non-negligible probability. This follows simply from the fact that when $\mathcal{A}_n$ does not output a good defense, the execution is truncated and the distributions generated by Sim and Sim' are *identical*. Therefore, Eq. (15) implies that for infinitely many n's, $\mathcal{A}_n$ outputs a good defense with probability at least $1/p(n)$. Next, recall that $\tilde{\mathcal{A}}_n$ chooses the sets Q and Q' randomly (under the constraints prescribed in the protocol). Thus, with probability exactly $1/4$, $j \in Q$ and $j \notin Q'$ (because the probability that a given j is in a specified set is exactly $1/2$). We conclude that with non-negligible probability, $\tilde{\mathcal{A}}_n$ does not output fail, and thus $\Pr[\mathsf{Expt}_{\pi}^{\mathrm{rec}}(\tilde{\mathcal{A}}_n) = 1]$ is non-negligible.

It remains to show that Sim runs in expected polynomial-time. Aside from the rewinding stage, all work takes a fixed polynomial amount of time. Regarding the rewinding stage, we have the following. Let p denote the probability that $\mathcal{A}_n$ replies correctly upon a random set of indices Q of size n, as specified in the protocol. Then, given that $\mathcal{A}_n$ replied correctly to the initial query set Q, the expected number of rewinding attempts with independent Q' made by Sim equals $1/p$. Since these rewinding attempts are only made if $\mathcal{A}_n$ replied correctly to the initial query set Q, we have that the expected number of attempts overall equals $p \cdot 1/p = 1$. This completes the proof. ■

5 Security for Malicious Senders and Privacy for Defensible Receivers

In this section, we *reverse* the oblivious transfer protocol of Protocol 4.2 to obtain a protocol that is secure in the presence of a malicious sender and private for random inputs in the presence of a defensible receiver. We use the construction of [38] for reversing Protocol 4.2. The protocol is as follows:

Protocol 5.1 (reversing oblivious transfer):

- **Inputs:** *The sender S has a pair of bits (s_0, s_1) for input and the receiver R has a bit r.*
- **The protocol:**
  1. *The sender and receiver run an oblivious transfer protocol π that is secure in the presence of a malicious receiver and private in the presence of a defensible sender:*
     (a) *The sender S, playing the receiver in π, inputs $\tilde{r} = s_0 \oplus s_1$*
     (b) *The receiver R, playing the sender in π, chooses a random bit $\rho \in_R \{0,1\}$ and inputs $\tilde{s}_0 = \rho$ and $\tilde{s}_1 = \rho \oplus r$.*

     Denote S's output from π by a.
  2. *S sends R the bit $\alpha = s_0 \oplus a$.*
  3. *R outputs $s_r = \rho \oplus \alpha$.*

We now prove the security of the reversed protocol. We remark that although Protocol 5.1 can easily be proven as an information-theoretic reduction (and likewise, when the original oblivious transfer protocol is secure in the presence of both a malicious sender and receiver), it is far more subtle in the setting where only privacy in the presence of a defensible sender is assumed.

Claim 5.2 *If π is a non-trivial oblivious transfer protocol that is secure in the presence of a malicious receiver and private in the presence of a defensible sender, then Protocol 5.1 is a non-trivial oblivious transfer protocol that is secure in the presence of a malicious sender and private for random inputs in the presence of a defensible receiver.*

Proof: We first show that if S and R follow the protocol, then R receives the correct output. In order to see this, notice that $\rho \oplus \alpha = \rho \oplus s_0 \oplus a$. Now, there are two cases:

1. *Case $s_0 \oplus s_1 = 0$:* In this case, $a = \rho$ and so $\rho \oplus \alpha = \rho \oplus s_0 \oplus \rho = s_0$. Now, since $s_0 \oplus s_1 = 0$, it holds that $s_0 = s_1$ and so R's output is correct (irrespective of the value of r).

2. *Case $s_0 \oplus s_1 = 1$:* In this case, $a = \rho \oplus r$ and so $\rho \oplus \alpha = \rho \oplus s_0 \oplus \rho \oplus r = s_0 \oplus r$. Now, on the one hand, if $r = 0$ then R outputs s_0, which is correct. On the other hand, if $r = 1$, then R outputs $s_0 \oplus 1$. Since, in this case, $s_0 \oplus s_1 = 1$, it holds that $s_1 = s_0 \oplus 1$, and so once again, R's output is correct.

We now proceed to prove the security and privacy of the protocol.

Security in the presence of a malicious sender. As in the proof of Claim 4.3, we present the proof in the "hybrid model", where the parties have access to a trusted party computing the oblivious transfer protocol.[9] Let $\mathcal{A}$ be an adversary who controls the sender S in Protocol 5.1. We construct a simulator Sim as follows:

1. Sim invokes $\mathcal{A}$ and receives from it the input $\sigma = s_0 \oplus s_1$ that $\mathcal{A}$ sends to the trusted party for the oblivious transfer subprotocol.
2. Sim hands $\mathcal{A}$ a uniformly distributed bit $a \in_R \{0,1\}$.
3. Sim receives a bit α from $\mathcal{A}$ and computes $s_0 = \alpha \oplus a$ and $s_1 = \alpha \oplus a \oplus \sigma$, and sends (s_0, s_1) to its own trusted party (recall that Sim works in the ideal model).
4. Sim outputs whatever $\mathcal{A}$ outputs.

We show that the joint output of Sim and the honest R (in the ideal model) is identical to the joint output of $\mathcal{A}$ and the honest R in a real execution of Protocol 5.1. First, note that the view of $\mathcal{A}$ in the simulation by Sim is identical to its view in a real execution; this follows simply from the fact that a is uniformly distributed (it equals either ρ or $\rho \oplus r$ where ρ is uniformly distributed). Next, note that R's output is fully determined by σ, a, and α. To see this, consider the following case analysis:

1. *Case $\sigma = 0$:* In this case, the output a that $\mathcal{A}$ receives in π equals ρ and so R's output equals $\rho \oplus \alpha = a \oplus \alpha$. Note that when $\sigma = 0$, it holds that $s_0 = s_1$. Therefore,the fact that Sim sends $s_0 = a \oplus \alpha$ and $s_1 = a \oplus \alpha \oplus \sigma = a \oplus \alpha = s_0$ results in the correct output distribution for R.
2. *Case $\sigma = 1$:* In this case, the output a that $\mathcal{A}$ receives in π equals $\rho \oplus r$ and so R's output equals $\rho \oplus \alpha = a \oplus r \oplus \alpha$ (because $\rho = a \oplus r$). Now, if $r = 0$, then R's output equals $a \oplus \alpha$ which is exactly the value of s_0 sent by Sim to the trusted party. On the other hand, if $r = 1$, then R's output equals $a \oplus 1 \oplus \alpha$. However, since $\sigma = 1$ in this case, we have that $a \oplus 1 \oplus \alpha = a \oplus \sigma \oplus \alpha$ which is exactly the value of s_1 sent by Sim to the trusted party.

We conclude that in all cases, the value that the honest R would output in a real execution of Protocol 5.1 with $\mathcal{A}$ equals the value that the honest R outputs in an ideal execution with Sim. Thus, the distribution generated by Sim in the ideal model is identical to a real execution of Protocol 5.1.

[9] We note that the composition theorems of [4, 13] both relate to the case of "full security". However, the analysis (for the case of static adversaries) works by separately analyzing a corrupted P_1 and a corrupted P_2. It therefore follows also in our case.

Privacy for random inputs in the presence of a defensible receiver. Let $\mathcal{A}$ be an adversary that controls R in an execution of Protocol 5.1. We show that $\mathcal{A}$ can learn at most one bit of (s_0, s_1). To see this, recall that π is private in the presence of a defensible sender. Thus, if $\mathcal{A}$ provides a good defense, it learns nothing of the sender's input $s_0 \oplus s_1$ into π. This implies that the only information that $\mathcal{A}$ learns from the protocol is $\alpha = s_0 \oplus a$. Intuitively, since a is essentially a function of $s_0 \oplus s_1$, the value α cannot reveal more than either s_0 or s_1.

To prove this formally, denote $r' = s_0 \oplus s_1$; recall that s_0 and s_1 are uniformly distributed, and therefore so is r'. Now, S's input to π is r', and following the execution, S sends R the message $\alpha = s_0 \oplus a$, where s_0 is uniformly distributed and *independent* of r' (independence follows because s_0 and s_1 are independent of each other and $r' = s_0 \oplus s_1$).[10] Thus, the message α that S sends to $\mathcal{A}$ controlling R can be simulated. This enables us to show that if a defensible receiver $\mathcal{A}$ can guess the values of both s_0 and s_1 with non-negligible advantage in an execution of Protocol 5.1, then this can be used by a defensible sender to guess the value of $r' = s_0 \oplus s_1$ with non-negligible advantage in an execution of π. This would then contradict the security of π.

Formally, assume by contradiction that there exists an adversarial defensible receiver $\mathcal{A} = \{\mathcal{A}_n\}$, a polynomial $p(\cdot)$ and infinitely many n's such that

$$\Pr\left[\mathsf{Expt}^{\mathrm{rec}}_{\Pi}(\mathcal{A}_n) = 1\right] \geq \frac{1}{2} + \frac{1}{p(n)}$$

where Π denotes Protocol 5.1 (not to be confused with the subprotocol π). We construct an adversarial defensible *sender* $\hat{\mathcal{A}}$ who attacks an execution of the subprotocol π, while interacting with an honest receiver R. Adversary $\hat{\mathcal{A}}$ internally invokes $\mathcal{A}$ and forwards all of $\mathcal{A}$'s messages to the honest receiver and vice versa. After the execution of π concludes, $\hat{\mathcal{A}}$ chooses a random bit $\alpha \in_R \{0,1\}$ and internally hands $\mathcal{A}$ the bit α. Adversary $\hat{\mathcal{A}}$ obtains $\mathcal{A}$'s defense and guess τ. If the defense is not good it outputs $\perp$. Otherwise, it uses the defense and τ to obtain a pair (s'_0, s'_1) as follows. $\mathcal{A}$'s defense represents an honest receiver's view in Protocol 5.1. As such, it contains an input r' and defines an output $s'_{r'}$ (a party's view implicitly contains its output). In addition, the bit τ represents $\mathcal{A}$'s guess $s'_{1-r'}$ for the honest sender's other input bit. Adversary $\hat{\mathcal{A}}$ defines its output guess to be $\tau' = s'_0 \oplus s'_1$. Furthermore, its defense just consists of the part of $\mathcal{A}$'s defense that relates to π. Adversary $\hat{\mathcal{A}}$ outputs this defense and τ'.

We now show that for some polynomial $p'(\cdot)$ and infinitely many n's

$$\left|\Pr\left[\mathsf{Expt}^{\mathrm{snd}}_{\pi}(\hat{\mathcal{A}}_n, 1) = 1\right] - \Pr\left[\mathsf{Expt}^{\mathrm{snd}}_{\pi}(\hat{\mathcal{A}}_n, 0) = 1\right]\right| \geq \frac{1}{p'(n)}$$

First, we claim that $\mathcal{A}$'s view in this simulation by $\hat{\mathcal{A}}$ is *identical* to its view in a real execution of Protocol 5.1 where $s_0, s_1 \in_R \{0,1\}$. This follows from the fact that α is uniformly distributed and independent of $s_0 \oplus s_1$. Furthermore, the input value used by the honest receiver running π with $\hat{\mathcal{A}}$ is distributed exactly according to $s_0 \oplus s_1$ (because they are both uniform). Finally, the output guess generated by $\hat{\mathcal{A}}$ is correct in the case that $\mathcal{A}$'s defense is good and τ really equals s_{1-r}. This holds because if $\mathcal{A}$'s defense is good, then the output it implicitly defines must equal s_r (because

[10] We stress an important point here: the assumption regarding the security of π is only that it is private for a defensible sender. Thus, there is no guarantee that the output a that S receives from π is "correctly" distributed. Nevertheless, S's input to π is r' and this is independent of s_0. Thus, α is uniformly distributed, irrespective of the output that S receives in π.

an honest receiver with input r must receive s_r). Thus, both s_0 and s_1 are correct in this case, and so is $s_0 \oplus s_1$. This implies that:

$$\Pr\left[\mathsf{Expt}_{\pi}^{\mathrm{snd}}(\hat{\mathcal{A}}_n, b) = b\right] = \Pr\left[\mathsf{Expt}_{\Pi}^{\mathrm{rec}}(\mathcal{A}_n) = 1\right] \geq \frac{1}{2} + \frac{1}{p(n)} \tag{16}$$

We wish to now bound $\Pr\left[\mathsf{Expt}_{\pi}^{\mathrm{snd}}(\hat{\mathcal{A}}_n, b) = 1 - b\right]$. This event (the output being $1-b$) can occur in the case that $\hat{\mathcal{A}}_n$'s defense is good, but its guess $\tau \neq b$. Denote by good the event that $\hat{\mathcal{A}}_n$'s defense is good, and by guess its guess-bit τ. Then,

$$\begin{aligned}
\Pr\left[\mathsf{Expt}_{\pi}^{\mathrm{snd}}(\hat{\mathcal{A}}_n, b) = 1 - b\right] &= \Pr\left[\mathsf{guess}(\mathsf{Expt}_{\pi}^{\mathrm{snd}}(\hat{\mathcal{A}}_n, b)) = 1 - b \;\&\; \mathsf{good}\right] \\
&= \Pr\left[\mathsf{guess}(\mathsf{Expt}_{\pi}^{\mathrm{snd}}(\hat{\mathcal{A}}_n, b)) = 1 - b \mid \mathsf{good}\right] \cdot \Pr[\mathsf{good}] \\
&= \left(1 - \Pr\left[\mathsf{guess}(\mathsf{Expt}_{\pi}^{\mathrm{snd}}(\hat{\mathcal{A}}_n, b)) = b \mid \mathsf{good}\right]\right) \cdot \Pr[\mathsf{good}] \\
&= \Pr[\mathsf{good}] - \Pr\left[\mathsf{guess}(\mathsf{Expt}_{\pi}^{\mathrm{snd}}(\hat{\mathcal{A}}_n, b)) = b \;\&\; \mathsf{good}\right] \\
&\leq \Pr[\mathsf{good}] - \frac{1}{2} - \frac{1}{p(n)}
\end{aligned}$$

where the last inequality follows from Eq. (16). Now, just by noting that $\Pr[\mathsf{good}] \leq 1$, we have that

$$\Pr\left[\mathsf{Expt}_{\pi}^{\mathrm{snd}}(\hat{\mathcal{A}}_n, b) = 1 - b\right] \leq 1 - \frac{1}{2} - \frac{1}{p(n)} = \frac{1}{2} - \frac{1}{p(n)} \tag{17}$$

Combining Equations (16) and (17), we have:

$$\left|\Pr\left[\mathsf{Expt}_{\pi}^{\mathrm{snd}}(\hat{\mathcal{A}}_n, 1) = 1\right] - \Pr\left[\mathsf{Expt}_{\pi}^{\mathrm{snd}}(\hat{\mathcal{A}}_n, 0) = 1\right]\right| \geq \frac{1}{2} + \frac{1}{p(n)} - \left(\frac{1}{2} - \frac{1}{p(n)}\right) = \frac{2}{p(n)}$$

in contradiction to the assumption that π is private in the presence of malicious receivers. ■

We have the following corollary:

Corollary 5.3 *There exists a fully black-box reduction from oblivious transfer that is secure in the presence of a malicious sender and private for random inputs in the presence of a defensible receiver, to oblivious transfer that is secure in the presence of a malicious receiver and private in the presence of a defensible sender.*

6 Fully Secure Oblivious Transfer

In this section, we use the construction of Protocol 4.2 again in order to boost the security of Protocol 5.1 so that it is secure in the presence of both a malicious sender and a malicious receiver; the resulting protocol satisfies the standard simulation-based security definition of oblivious transfer (see Definition 2.1).

By Claim 4.3, we have that Protocol 4.2 boosts the security of *any* oblivious transfer protocol that is private for defensible receivers into one that is secure in the presence of malicious receivers. We can therefore use Protocol 4.2 to boost the security of Protocol 5.1 so that the result is a protocol that is secure in the presence of malicious receivers. This does not suffice, however, because we

must show that if the subprotocol used in Protocol 4.2 is *secure* in the presence of malicious senders, then the result is still secure in the presence of malicious senders. (Claim 4.2 considers only privacy for defensible senders.) This is actually easy to show, and we sketch the proof here.

Theorem 6.1 *There exists a fully black-box reduction from fully secure oblivious transfer to oblivious transfer that is secure in the presence of malicious senders and private for random inputs in the presence of defensible receivers.*

Proof Sketch: Let π be any oblivious transfer protocol that is secure in the presence of malicious senders and private for random inputs in the presence of defensible receivers, and let π' be the result of applying the transformation of Protocol 4.2 to this π. We claim that π' is fully secure.

Since π is private for random inputs in the presence of defensible receivers, Claim 4.3 immediately implies that π' is secure in the presence of malicious receivers. It remains to show that security in the presence of malicious senders is preserved. We construct the following simulator Sim for a malicious sender (since the subprotocol π is secure in the presence of a malicious sender, $\mathcal{A}$ works in the "hybrid model" with access to a trusted party computing the oblivious transfer functionality):

1. Sim invokes $\mathcal{A}$ and plays the honest receiver using input bit $r = 0$.
2. Since $\mathcal{A}$ works in the hybrid model, Sim obtains all of the pairs s_i^0, s_i^1 that $\mathcal{A}$ inputs into the oblivious transfer protocol.
3. At the conclusion of the protocol, Sim obtains the values σ_0 and σ_1. Since Sim knows all of the s_i^b values, it can compute (s_0, s_1). Therefore, Sim computes these values and sends them to the trusted party. This completes the simulation.

First, the view of $\mathcal{A}$ in the simulation is computationally indistinguishable from its view in a real execution. This follows from the fact that it learns nothing of the r_i values in the oblivious transfer subprotocol. Thus, the $\alpha_j = r \oplus r_j$ values reveal nothing about r. Next, we claim that the joint output of $\mathcal{A}$ and the honest receiver in the ideal model is indistinguishable from the joint output in a real execution. This follows from the fact that the honest receiver's output is determined from the σ_0 and σ_1 values (irrespective of whether or not $\mathcal{A}$ computes them "correctly"). Since Sim obtains these values in the same way as the honest receiver would (with the only difference that it can obtain both), we have that the receiver's output is the same in the real and ideal executions. This completes the proof sketch. ■

Conclusion – black-box construction of oblivious transfer. By combining Corollary 3.9, Theorem 4.1, Corollary 5.3, and Theorem 6.1, we have that there exists a fully black-box reduction from *fully secure oblivious transfer* to semi-honest oblivious transfer. In addition, there exist fully black-box reductions from semi-honest oblivious transfer to enhanced trapdoor permutations [9] and homomorphic encryption (cf. [1]). We therefore conclude with the following corollary:

Corollary 6.2 *There exists a fully black-box reduction from fully secure oblivious transfer to semi-honest oblivious transfer. Furthermore, there exist fully black-box reductions from fully secure oblivious transfer to enhanced trapdoor permutations and homomorphic encryption schemes.*

7 Black-Box Secure Multiparty Computation

Kilian [25], and later Ishai et al. [23], showed that any functionality can be securely computed given black-box access to an oblivious transfer functionality. We therefore have the following theorem, that constitutes our main result:

Theorem 7.1 *For any probabilistic polynomial-time functionality f, there exists a fully black-box reduction from a protocol that securely computes f (with any number of corrupted parties, in the presence of a static malicious adversary) to semi-honest oblivious transfer. Furthermore, there exist similar reductions to a family of enhanced trapdoor permutations and a homomorphic encryption scheme.*

Proof: In [25, 23] it is shown that f can be securely computed in the OT-hybrid model, i.e., in a model where the parties can access an ideal oblivious transfer oracle. Using a sequential composition theorem [4, 13, 14], we can (sequentially) replace each oracle call by an instance of the protocol from Corollary 6.2 (using either semi-honest oblivious transfer, enhanced trapdoor permutations, or homomorphic encryption) and obtain reductions as required. We note that Corollary 6.2 only guarantees security with *expected* polynomial-time simulation. However, this does not cause a problem because the straight-line simulation of the protocol from [23] guarantees its security even against expected polynomial-time adversaries (with expected polynomial-time simulation). This ensures (cf. [14]) that the resulting protocol for f is secure with expected polynomial-time simulation. ■

An important ramification of Theorem 7.1 is the following corollary:

Corollary 7.2 *For any probabilistic polynomial-time functionality f there exists a protocol that uses only black-box access to a family of enhanced trapdoor permutations or a homomorphic encryption scheme, and securely computes f with any number of corrupted parties and in the presence of a static malicious adversary.*

We remark that as is standard for the setting of no honest majority, the security guarantee achieved here is that of "security with abort"; see Definition 2.1 for a formal definition in the two-party case and [13, Chapter 7] for the general multiparty case.

References

[1] W. Aiello, Y. Ishai and O. Reingold. Priced Oblivious Transfer: How to Sell Digital Goods. In *EUROCRYPT 2001*, Springer-Verlag (LNCS 2045), pages 119–135, 2001.

[2] M. Ben-Or, S. Goldwasser and A. Wigderson. Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation. In 20*th STOC*, pages 1–10, 1988.

[3] M. Blum. Coin Flipping by Phone. In *IEEE Spring COMPCOM*, pages 133–137, 1982.

[4] R. Canetti. Security and Composition of Multiparty Cryptographic Protocols. *Journal of Cryptology*, 13(1):143–202, 2000.

[5] D. Chaum, C. Crépeau and I. Damgård. Multi-party Unconditionally Secure Protocols. In 20*th STOC*, pages 11–19, 1988.

[6] S. G. Choi, D. Dachman-Soled, T. Malkin, and H. Wee. Simple, Black-Box Constructions of Adaptively Secure Protocols. In *6th TCC*, pages 387-402, 2009.

[7] I. Damgård and Y. Ishai. Constant-Round Multiparty Computation Using a Black-Box Pseudorandom Generator. In *CRYPTO 2005*, Springer-Verlag (LNCS 3621), pages 378–394, 2005.

[8] D. Dolev, C. Dwork and M. Naor. Non-Malleable Cryptography. *SIAM Journal on Computing,* 30(2):391–437, 2000.

[9] S. Even, O. Goldreich and A. Lempel. A Randomized Protocol for Signing Contracts. In *Communications of the ACM,* 28(6):637–647, 1985.

[10] R. Gennaro and L. Trevisan. Lower Bounds on the Efficiency of Generic Cryptographic Constructions. In 41*st FOCS,* pages 305–314, 2000.

[11] Y. Gertner, S. Kannan, T. Malkin, O. Reingold and M. Viswanathan. The Relationship between Public Key Encryption and Oblivious Transfer. In 41*st FOCS,* pages 325–334, 2000.

[12] Y. Gertner, T. Malkin and O. Reingold. On the Impossibility of Basing Trapdoor Functions on Trapdoor Predicates. In 42*nd FOCS,* pages 126–135, 2001.

[13] O. Goldreich. *Foundations of Cryptography: Volume 2 – Basic Applications*. Cambridge University Press, 2004.

[14] O. Goldreich. On Expected Probabilistic Polynomial-Time Adversaries: A Suggestion for Restricted Definitions and Their Benefits. In the 4*th TCC*, Springer-Verlag (LNCS 4392), pages 174–193, 2007.

[15] O. Goldreich, S. Micali and A. Wigderson. Proofs that Yield Nothing but their Validity or All Languages in NP Have Zero-Knowledge Proof Systems. *Journal of the ACM*, 38(1):691–729, 1991.

[16] O. Goldreich, S. Micali and A. Wigderson. How to Play any Mental Game – A Completeness Theorem for Protocols with Honest Majority. In 19*th STOC,* pages 218–229, 1987. For details see [13, Chapter 7].

[17] I. Haitner. Semi-honest to Malicious Oblivious Transfer – the Black-Box Way. In the *5th TCC*, Springer-Verlag (LNCS 4948) pages 412–426, 2008.

[18] I. Haitner, M.-H. Nguyen, S. J. Ong, O. Reingold, and S. P. Vadhan. Statistically Hiding Commitments and Statistical Zero-Knowledge Arguments from Any One-Way Function. *SIAM Journal on Computing* 39(3):1153–1218, 2009.

[19] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[20] R. Impagliazzo and M Luby. One-way Functions are Essential for Complexity Based Cryptography. In *30th FOCS*, pages 230-235, 1989.

[21] R. Impagliazzo and S. Rudich. Limits on the Provable Consequences of One-way Permutations. In *CRYPTO'88*, Springer-Verlag (LNCS 403), pages 8–26, 1988.

[22] Y. Ishai, E. Kushilevitz, Y. Lindell and E. Petrank. Black-Box Constructions for Secure Multiparty Computation. In the 38*th STOC*, pages 99-108, 2006.

[23] Y. Ishai, M. Prabhakaran and A. Sahai. Founding Cryptography on Oblivious Transfer - Efficiently. In *CRYPTO 2008*, Springer-Verlag (LNCS 5157), pages 572–591, 2008.

[24] Y.T. Kalai. Smooth Projective Hashing and Two-Message Oblivious Transfer. In *EUROCRYPT 2005*, Springer-Verlag (LNCS 3494) pages 78–95, 2005.

[25] J. Kilian. Founding Cryptograph on Oblivious Transfer. In 20*th STOC*, pages 20–31, 1988.

[26] J. Kilian. *Uses of Randomness In Algorithms and Protocols.* MIT Press, 1990.

[27] J. Kilian. A Note on Efficient Zero-Knowledge Proofs and Arguments. In *24th STOC*, pages 723-732, 1992.

[28] J.H. Kim, D.R. Simon and P. Tetali. Limits on the Efficiency of One-Way Permutation-Based Hash Functions. In *40th FOCS*, pages 535–542, 1999.

[29] E. Kushilevitz and R. Ostrovsky. Replication Is Not Needed: Single Database, Computationally-Private Information Retrieval. In 38*th FOCS*, pages 364–373, 1997.

[30] Y. Lindell. A Simpler Construction of CCA2-Secure Public-Key Encryption Under General Assumptions. In *EUROCRYPT 2003,* Springer-Verlag (LNCS 2656), pages 241–254, 2003.

[31] S. Micali. Computationally Sound Proofs. *SIAM Journal on Computing*, 30(4):1253–1298, 2000.

[32] M. Naor. Bit Commitment Using Pseudorandomness. *J. Cryptology* 4(2): 151-158, 1991.

[33] M. Naor and K. Nissim. Communication Preserving Protocols for Secure Function Evaluation. In 33*rd STOC*, pages 590–599, 2001.

[34] M. Naor and B. Pinkas. Efficient Oblivious Transfer Protocols. In 12*th SODA*, pages 458–457, 2001.

[35] M. Rabin. How to Exchange Secrets by Oblivious Transfer. Tech. Memo TR-81, Aiken Computation Laboratory, Harvard U., 1981. Can be found at *Cryptology ePrint Archive*, Report 2005/187.

[36] O. Reingold, L. Trevisan, and S. Vadhan. Notions of Reducibility between Cryptographic Primitives. In 1*st TCC*, pages 1–20, 2004.

[37] A. Sahai. Non-Malleable Non-Interactive Zero-Knowledge and Adaptive Chosen-Ciphertext Security. In 40*th FOCS*, pages 543–553, 1999.

[38] S. Wolf and J. Wullschleger. Oblivious Transfer Is Symmetric. In *EUROCRYPT 2006*, Springer-Verlag (LNCS 4004), pages 222–232, 2006.

[39] A. Yao. How to Generate and Exchange Secrets. In 27*th FOCS*, pages 162–167, 1986.

A Standard Cryptographic Primitives

In this section we define one-way functions and commitment schemes, and note the relation between them.

A.1 One-way Functions

Definition A.1 *An polynomial-time computable function* $f\colon \{0,1\}^n \mapsto \{0,1\}^{\ell(n)}$ *is* one-way *if the following hold for every family of polynomial-size circuits* $A = \{A_n\}_{n\in\mathsf{N}}$*:*

$$\Pr_{y\leftarrow f(U_n)}[A_n(y) \in f^{-1}(y)] = \mathsf{neg}(n)$$

A.2 Commitment Schemes

Commitment schemes are the main tool we use in the transformation from semi-honest security to defensible security.

Definition A.2 *A* commitment scheme (S, R) *for (bit) strings of length* k *is an efficient two-party protocol consisting of two stages. Throughout, both parties receive the security parameter* 1^n *as input.*

> **Commit**. *The sender* S *has a private input* $\sigma \in \{0,1\}^k$*, which she wishes to commit to the receiver* R*, and a sequence of coin tosses* r*. At the end of this stage, both parties receive as common output a* commitment z*.*
>
> **Reveal**. *Both parties receive as input a commitment* z*.* S *also receives the private input* σ *and coin tosses* r *used in the commit stage. This stage is non-interactive:* S *sends a single message to* R*, and* R *either outputs a bit-string (and accepts) or rejects.*

In addition, the following properties hold:

> **(Perfect) Completeness**. *If both parties are honest, then for any string* $\sigma \in \{0,1\}^k$ *that* S *gets as private input,* R *accepts and outputs* σ *at the end of the reveal stage.*
>
> **(Computational) Hiding**. *For any family of polynomial-size circuits* $R^* = \{R^*\}_{n\in\mathsf{N}}$ *and any* $\sigma, \sigma' \in \{0,1\}^k$*, the distribution ensembles* $\mathsf{view}_{R_n^*}(S(1^n, \sigma), R_n^*)$ *and* $\mathsf{view}_{R_n^*}(S(1^n, \sigma'), R_n^*)$ *are computationally indistinguishable.*
>
> **(Computational) Binding**. *Any family of polynomial-size circuits* $S^* = \{S_n^*\}_{n\in\mathsf{N}}$ *succeeds in the following game (breaks the commitment) with only negligible probability in* n*:*
>
> - S_n^* *interacts with an honest* R *in the commit stage on security parameter* 1^n*, which yields a commitment* z*.*
> - S_n^* *outputs two messages* τ_0, τ_1 *and* distinct *strings* $\sigma_0, \sigma_1 \in \{0,1\}^k$*, such that for both* $b \in \{0,1\}$ *the receiver* R *on input* (z, τ_b) *accepts and outputs* σ_b*.*

It is well known that commitment scheme can be based on one-way functions in a black-box way.

Theorem A.3 *[[19, 32],[18]] There is a fully black-box construction of a commitment scheme from a one-way function, for any polynomial string length* $k(n)$*.*

Remarks:

Non-uniform security: To be consistent with rest of the paper, we defined the security of one-way functions and commitment schemes above with respect to non-uniform adversaries. While [19, 32] and [18] are stated in the uniform setting, their uniform proofs of security immediately yield the non-uniform variant stated in Theorem A.3.

Statistical security: We note that the commitment schemes achieved by [19, 32] and by [18], have stronger guarantees than stated in Theorem A.3 (statistical binding by [19, 32], and statistical hiding by [18]). For the sake of this paper, however, we only care about computational security.

B One-Way Functions from Semi-Honest Oblivious Transfer

In this appendix we show how one-way functions can be constructed in a black-box way from semi-honest oblivious transfer. It is possible to achieve this result by using the reduction from semi-honest oblivious transfer to key-agreement of [11], and then use the reduction of the latter to one-way functions by [20]. Since the following alternative proof is rather simple, we have chosen to give it here for the sake of self containment.

Theorem B.1 *There exists a fully black-box reduction from one-way functions to semi-honest oblivious transfer.*

Proof: We start by using semi-honest oblivious transfer to construct a semi-honest bit commitment scheme, where the latter is a bit commitment scheme that only guarantees security when both parties are acting semi-honestly in the *commit stage*. We stress that the sender is allowed to be malicious in the decommit phase, and binding must still hold. We then finish the proof by using a semi-honest bit commitment scheme to construct a one-way function.

Semi-honest oblivious transfer to semi-honest bit commitment. Let (S, R) be a semi-honest oblivious transfer protocol. We construct the semi-honest commitment (S', R') as follows:

Inputs: S' has a bit b to commit to; both parties also have input 1^n.

Commit stage: R' chooses uniformly at random n pairs of secrets $(\sigma_0^0, \sigma_0^1), \ldots, (\sigma_n^0, \sigma_n^1)$, and S' chooses a random index $i \in \{0, 1\}$. For each $j \in [n]$, the two parties run the semi-honest oblivious transfer protocol $(S(\sigma_j^0, \sigma_j^1), R(i))$, where R' plays the sender in the oblivious transfer and S' plays the receiver. S' sends $c = i \oplus b$ to R' as its commitment to b.

Reveal stage: S' sends i, b, and $\sigma_1^i, \ldots, \sigma_n^i$ to R'. R' accepts the decommitment, outputting b, if $i \oplus b = c$ and the values of $\sigma_1^i, \ldots, \sigma_n^i$ sent by S' are consistent with the values input by R' to the oblivious transfers.

The correctness of the above protocol is obvious. Furthermore, since the privacy of the oblivious transfer implies that the value of i does not leak to $S = R'$ (even in a polynomial number of executions), the hiding of the commitment scheme follows. Finally, the binding follows since in order to violate it, the sender S' needs to guess correctly the values of $\sigma_1^{1-i}, \ldots, \sigma_1^{1-i}$. By the privacy of the semi-honest oblivious transfer, however, the sender cannot do the latter with more than negligible success probability, as long as it behaved semi-honestly in the commit phase.

Semi-honest bit commitment to one-way functions. Given a semi-honest bit commitment scheme (S', R'), let b, $\rho_{S'}$ and $\rho_{R'}$ be the secret bit of S' and the random coins of the parties respectively. We define a function

$$f(1^n, b, \rho_{S'}, \rho_{R'}) \stackrel{\text{def}}{=} \mathsf{trans}(S'(1^n, b, \rho_{S'}), R'(1^n, \rho_{R'})).$$

Note that the f is not necessarily defined over all input lengths. Nevertheless, we show that f is hard to invert over the infinite set of input lengths implied by the above definition. The existence of a one-way function which is defined over all input lengths follows by using standard padding techniques. (In particular, given a string of any length n, the first step is to take the longest prefix of the string that can be interpreted as $(b, \rho_{S'}, \rho_{R'})$. Then f is applied to 1^n and this prefix.)

Let $m(n)$ be the total length of the random coins of the parties executed with security parameter 1^n and assume the existence of a probabilistic polynomial-time A such that following is non negligible

$$\Pr_{b \leftarrow \{0,1\}, x \leftarrow \{0,1\}^{m(n)}} \left[A(1^n, f(b,x) \in f^{-1}(f(b,x))) \right].$$

We first claim that the probability that A outputs a valid preimage containing a *different* b is negligible. That is,

$$\Pr_{b \leftarrow \{0,1\}, x \leftarrow \{0,1\}^{m(n)}} \left[(b', X_S) \leftarrow A(1^n, f(x) \mid (b', X_S) \in f^{-1}(f(b,x))) \wedge b' \neq b \right] = \mathsf{neg}(n) \ .$$

Assume otherwise and consider a semi-honest sender that after the execution of the protocol runs A on the transcript. With non-negligible probability, the sender obtains a pair $(b', \rho'_{S'}, \rho'_{R'})$ such that $b \neq b'$ and $S'(b', \rho'_{S'})$ is consistent with the protocol. This is a contradiction, since the above sender can violate the binding of the commitment.

Consider now the semi-honest receiver that after the interaction concludes runs A to invert f, obtaining (b', X_S). The receiver checks if (b', X_S) is consistent with the transcript of the protocol. If yes, it outputs b'; otherwise it outputs a random bit. Since $b' = b$ except with negligible probability (as we have shown) it follows that the receiver has a non-negligible advantage in guessing b, thereby violating the hiding of the semi-honest commitment. ■